CageControl

Control waveplates inside tomography cages

Generated by Doxygen 1.8.13

Contents

Chapter 1

Bug List

File debug.h

Printing to console does not work on Windows. Workaround: Redirect stderr to stdout and redirect stdout to a file.

File defines.h

There are no known bugs.

Namespace helper

There are no known bugs.

Class Motor

There are no known bugs.

Class UDPlistener

There are no known bugs

File version.h

There are no known bugs.

Chapter 2

Namespace Index

2.1 Namespace List

Here is a list of all documented namespaces with brief descriptions:

Chapter 3

Hierarchical Index

3.1 Class Hierarchy

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

Chapter 4

Class Index

4.1 Class List

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

Chapter 5

File Index

5.1 File List

Here is a list of all documented files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 helper Namespace Reference

contains small functions to display messages

Functions

- void message (QString msg)
  message displays a message box
- void error (QString msg)
  error displays an error-messagebox and writes a debug_error message to stdout
- void warning (QString msg)
  warning displays warning-messagebox and writes a debug_warning message to stdout
- void info (QString msg)
  info displays an info-messagebox and writes a debug_info message to stdout

6.1.1 Detailed Description

contains small functions to display messages

Bug There are no known bugs.

6.1.2 Function Documentation

6.1.2.1 error()

```
void helper::error (
            QString msg )
```

error displays an error-messagebox and writes a debug_error message to stdout

Parameters

msg	the message to be displayed

6.1.2.2 info()

```
void helper::info (
            QString msg )
```

info displays an info-messagebox and writes a debug_info message to stdout

Parameters

msg	the message to be displayed

6.1.2.3 message()

```
void helper::message (
            QString msg )
```

message displays a message box

Parameters

msg	the message to be displayed

6.1.2.4 warning()

```
void helper::warning (
            QString msg )
```

warning displays warning-messagebox and writes a debug_warning message to stdout

Parameters

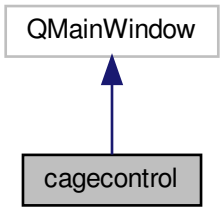
msg	the message to be displayed

Chapter 7

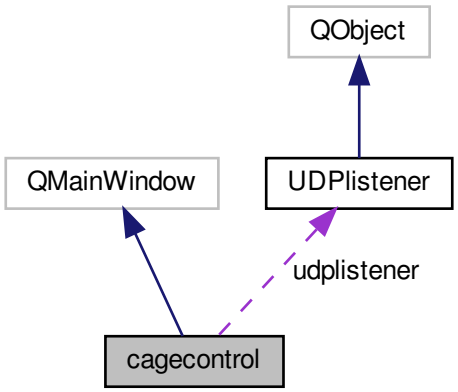
Class Documentation

7.1 cagecontrol Class Reference

Inheritance diagram for cagecontrol:



Collaboration diagram for cagecontrol:



Public Slots

- void slot_changeWPangles (QVector< double > angles)

  slot_changeWPangles sets offsetangles for all waveplates
- void slot_changeoffsetusage (bool uo_in)

  slot_changeoffsetusage changes the usage of the waveplate offset
- void slot_moveHV (QString color)

  moveHV
- void slot_movePM (QString color)

  movePM
- void slot_moveLR (QString color)

  moveLR
- void slot_movemotors (QString color, double HWPang, double QWPang)

  movemotors

Public Member Functions

- **cagecontrol** (QWidget *parent=nullptr)

Private Slots

- void updatesettings (double d)

  updatesettings fills variables with data from GUI
- void updatesettingsint (int i)

  updatesettingsint wrapper, just calls
- void updateUI ()

  updateUI updates UI with supposedly new numbers (loaded from conf file, e.g.)

Private Member Functions

- void setupUI (QGridLayout *layout)

  Puts together the GUI.
- void openmotors ()

  Opens serial connections to the PCB motor controlllers.
- void updatestatus (QString msg)

  updatestatus writes message to statusbar and to a logfile
- void LoadConfig ()

  LoadConfig loads config from a file.
- void SaveConfig ()

  SaveConfig stores config to a file.
- void motorGB (QGroupBox *gb, QString id)

  motorGB fills an empty QGroupBox with motor controls
- void initconnections ()

  initconnections connects Qt Signals to slots
- void movemotor (QString motor, double HWPang, double QWPang)

  movemotor moves both motors of a cage to certain angles
- void moveredHV ()

  moveredHV moves red cage to HV basis
- void moveredPM ()

moveredPM moves red cage to PM basis

- void moveredLR ()

  moveredLR moves red cage to RL basis
- void moveredANG ()

  moveredANG moves red cage to the angles set in the GUI
- void movebrownHV ()

  movebrownHV moves brown cage to HV basis
- void movebrownPM ()

  movebrownPM moves brown cage to PM basis
- void movebrownLR ()

  movebrownLR moves brown cage to RL basis
- void movebrownANG ()

  movebrownANG moves brown cage to the angles set in the GUI
- void movegreenHV ()

  movegreenHV moves green cage to HV basis
- void movegreenPM ()

  movegreenPM moves green cage to PM basis
- void movegreenLR ()

  movegreenLR moves green cage to RL basis
- void movegreenANG ()

  movegreenANG moves green cage to the angles set in the GUI
- void moveblueHV ()

  moveblueHV moves blue cage to HV basis
- void movebluePM ()

  movebluePM moves blue cage to PM basis
- void moveblueLR ()

  moveblueLR moves blue cage to RL basis
- void moveblueANG ()

  moveblueANG moves white cage to the angles set in the GUI
- void movewhiteHV ()

  movewhiteHV moves white cage to HV basis
- void movewhitePM ()

  movewhitePM moves white cage to PM basis
- void movewhiteLR ()

  movewhiteLR moves white cage to RL basis
- void movewhiteANG ()

  movewhiteANG moves black cage to the angles set in the GUI
- void moveblackHV ()

  moveblackHV moves black cage to HV basis
- void moveblackPM ()

  moveblackPM moves black cage to PM basis
- void moveblackLR ()

  moveblackLR moves black cage to RL basis
- void moveblackANG ()

  moveblackANG
- void moveallhv ()

  moveallhv
- void moveallpm ()

  moveallpm
- void movealllr ()

  movealllr
- void moveallarb ()

  moveallarb

Private Attributes

- int udpport

  Hold the UDP port to listen to for commandds.
- bool pauseupdating

  Keep updateUI and updatesettings from interfering with each other.
- bool useoffset

  If true, the angles in the settings-tab will be used as '0'.
- QSettings * settings

  A QSettings object, used to store settings in a config file.
- UDPlistener * udplistener

  Listens to a UDP port, aquiires & checks commands send to it.
- QTabWidget * tabs

  GUI tab widget.
- QWidget * settingstab

  GUI tab containing settings.
- QWidget * motorstab

  GUI tab containing motor controls.
- QStatusBar * status

  Status bar.
- QVector< QString > comports

  Vector containing available serial ports names ports.
- QVector< bool > invert

  True: invert predefined bases (H/V -> V/H, P/M->M/P, L/R->R/L)
- QVector< Motor * > motors

  List of serial connections to the cages.
- QVector< QString > motorName

  List of colorcodes of the cages.
- QVector< QDoubleSpinBox > HWP0sp

  List of QSpinBoxes to set the '0' of the HWPs.
- QVector< QDoubleSpinBox > QWP0sp

  List of QSpinBoxes to set the '0' of the QWPs.
- QVector< int > HWPmnum

  Motornumber of controller the HWP is connected to.
- QVector< int > QWPmnum

  Motornumber of controller the QWP is connected to.
- QVector< double > HWP0

  '0' of HWPs
- QVector< double > QWP0

  '0' of QWPs
- QVector< double > HWPcust

  custum set angle to rotate HWP to
- QVector< double > QWPcust

  custom set angle to rotate QWP to
- QVector< QGroupBox * > uiMotorGroupBoxes

  List of Groupboxes containing cage controls.

7.1.1 Member Function Documentation

7.1.1.1 initconnections()

```
void cagecontrol::initconnections ( )  [private]
```

initconnections connects Qt Signals to slots

Defines what happens when a button is clicked, a number is changet, et cetera

7.1.1.2 LoadConfig()

```
void cagecontrol::LoadConfig ( )  [private]
```

LoadConfig loads config from a file.

The dialog is set up with values already stored in the QSettings object. If a specific quantity does not exist there, it is set to a standard value.

7.1.1.3 motorGB()

```
void cagecontrol::motorGB (
            QGroupBox * gb,
            QString id )  [private]
```

motorGB fills an empty QGroupBox with motor controls

Parameters

gb	empty QGroupBox
id	colorcode of the cage

7.1.1.4 movemotor()

```
void cagecontrol::movemotor (
            QString motor,
            double HWPang,
            double QWPang )  [private]
```

movemotor moves both motors of a cage to certain angles

Parameters

motor	colorcode of the cage
HWPang	angle of the HWP in degrees
QWPang	angle of the QWP in degrees

7.1.1.5 SaveConfig()

```
void cagecontrol::SaveConfig ( )  [private]
```

SaveConfig stores config to a file.

The QSettings object is updated with the values received from the dialog and saved immediately.

7.1.1.6 slot_changeoffsetusage

```
void cagecontrol::slot_changeoffsetusage (
            bool uo_in )  [slot]
```

slot_changeoffsetusage changes the usage of the waveplate offset

Parameters

useoffset	true if waveplate offset is to be used

The term 'offset' refers to the waveplate angles specified in the

See also

settingstab. E.g.: 'H' of HWP specified in he settingstab is 50° and one wants to rotate the waveplate to H+10°. If (useoffset==true), one needs to rotate the motor to 10°. If (useoffset==false), one needs to rotate the motor to 60°.

7.1.1.7 slot_changeWPangles

```
void cagecontrol::slot_changeWPangles (
            QVector< double > angles )  [slot]
```

slot_changeWPangles sets offsetangles for all waveplates

Parameters

angles	vector containing all angles. Ordering: HWP0,HWP1,...,HWPn,QWP0,QWP1,...,QWPn

7.1.1.8 slot_moveHV

```
void cagecontrol::slot_moveHV (
            QString color )  [slot]
```

moveHV

Parameters

color	colorcode of the cage, or 'all'

7.1.1.9 slot_moveLR

```
void cagecontrol::slot_moveLR (
            QString color )  [slot]
```

moveLR

Parameters

color	colorcode of the cage, or 'all'

7.1.1.10 slot_movemotors

```
void cagecontrol::slot_movemotors (
            QString color,
            double HWPang,
            double QWPang )  [slot]
```

movemotors

Parameters

color	colorcode of the cage, or 'all'

7.1.1.11 slot_movePM

```
void cagecontrol::slot_movePM (
            QString color )  [slot]
```

movePM

Parameters

color	colorcode of the cage, or 'all'

7.1.1.12 updatesettings

```
void cagecontrol::updatesettings (
            double d )  [private], [slot]
```

updatesettings fills variables with data from GUI

Parameters

d	unused

7.1.1.13 updatesettingsint

```
void cagecontrol::updatesettingsint (
            int i )  [private], [slot]
```

updatesettingsint wrapper, just calls

See also

updatesettings(double d)

Parameters

i	unused

7.1.1.14 updatestatus()

```
void cagecontrol::updatestatus (
            QString msg )  [private]
```

updatestatus writes message to statusbar and to a logfile

Parameters

msg	Message to write

7.1.2 Member Data Documentation

7.1.2.1 useoffset

```
bool cagecontrol::useoffset  [private]
```

If true, the angles in the settings-tab will be used as '0'.

See also

slot_changeoffsetusage

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

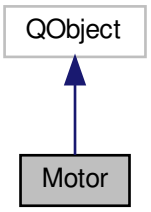
- cagecontrol.h
- cagecontrol.cpp
- cagecontrol_motors.cpp
- cagecontrol_ui.cpp

7.2 Motor Class Reference

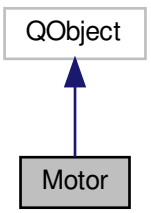
The Motor class operates the PCB-motor.

```
#include <motor.h>
```

Inheritance diagram for Motor:



Collaboration diagram for Motor:



Public Slots

- void open (QString port)

  open establishes a connection over a serial port
- void close ()

  close closes the serialport connection
- void read ()

  read reads from the serial port
- void write (const QByteArray &data)

  write writes to the serialport
- void handleError (QSerialPort::SerialPortError error)

  handleError prints an error message of the serialport connection and closes the connection
- void showStatusMessage (const QString &message)

  showStatusMessage fills the label in the GUI with text
- bool isopen ()

  isopen returns the state of the serial connection
- void command_park ()

  command_park moves the motor to the mechanical stop
- void command_home ()

  command_home sends commands to position at the mechanical stop and afterwards go to the offset starting position, but in an inaccurate way
- void command_info ()

  command_info sends the command to request the PCBMotor information
- void command_help ()

  command_help sends the command to print the PCBMotor help
- void command_frequency_sweep ()

  command_frequency_sweep sends the PCBMotor command for a frequency sweep
- void command_singlestep (QString dirstring)

  command_singlestep moves the motor a single step in a direction specified by dirstring
- void command_step (uint16_t numsteps, QString dirstring)

  command_step moves the motor numstep steps in a direction specified by dirstring
- void command_microstep (uint16_t nummsteps, QString dirstring)

  command_microstep aplies nummsteps micropulses to the motor
- void stop (bool stop)

  stop Tries to stop movenents if possible
- void command_moveboth (double ang1, double ang2)

  command_moveboth moves both motors connected to the controller

Signals

- void motorstatusmessage (const QString &message)

  motorstatusmessage emitted when the status of the serial connection changes, with a string indicating the actual state.
- void ConnectionClosed ()

  emitted when serial connection is closed

Public Member Functions

- Motor ()

  Motor the contructor initializes variables and establishes the serial connection.
- bool sensordata ()

  sensordata returns the current PCBMotor optical encoder wheel sendor state

Public Attributes

- QString publicmotorstatusmessage

  A string containing the current state of the serial connection.
- QSerialPort * serial

  Qt serial connection interface.

Private Member Functions

- void moveboth ()

  command_moveboth moves both motors connected to the controller

Private Attributes

- QTimer hometimer

  Used to iterate through the steps of 'go to the starting position' - but in an inaccurate way.
- QTimer bothtimer

  Used to iterate through the steps of moving two motors of one controller.
- int movebothstep

  Controls logic flow when two motors are to be moved consecutively.
- bool serialconnectionok

  False if opening the serial connection failed.
- uint16_t motor1steps

  number of stept the 1st motor is to be moved
- uint16_t motor2steps

  number of stept the 2nd motor is to be moved

7.2.1 Detailed Description

The Motor class operates the PCB-motor.

Bug There are no known bugs.

The PCBMotor is controllable by sending ASCII commands over a serial connection. This class establishes such a connection and controls the movements of the motor.

7.2.2 Member Function Documentation

7.2.2.1 command_microstep

```
void Motor::command_microstep (
            uint16_t nummsteps,
            QString dirstring )  [slot]
```

command_microstep aplies nummsteps micropulses to the motor

Parameters

nummsteps	number of micropulses to apply
dirstring	string containing the desired direction

dirstring may either be "bw" of "fw" for backward/forward movement.

7.2.2.2 command_moveboth

```
void Motor::command_moveboth (
            double ang1,
            double ang2 )  [slot]
```

command_moveboth moves both motors connected to the controller

Parameters

ang1	angle motor 1 is to be moved to
ang2angle	motor 2 is to be moved to

7.2.2.3 command_singlestep

```
void Motor::command_singlestep (
            QString dirstring )  [slot]
```

command_singlestep moves the motor a single step in a direction specified by dirstring

Parameters

dirstring	a string containing the desired movenent direction

Dirstring may either be "bw" of "fw" for backward/forward movement.

7.2.2.4 command_step

```
void Motor::command_step (
            uint16_t numsteps,
            QString dirstring )  [slot]
```

command_step moves the motor numstep steps in a direction specified by dirstring

Parameters

numsteps	number of steps to go
dirstring	direction to go

Dirstring may either be "bw" of "fw" for backward/forward movement.

7.2.2.5 handleError

```
void Motor::handleError (
            QSerialPort::SerialPortError error )  [slot]
```

handleError prints an error message of the serialport connection and closes the connection

Parameters

error	

7.2.2.6 isopen

```
bool Motor::isopen ( )  [slot]
```

isopen returns the state of the serial connection

Returns

true if serial connection was established successfully, false otherwise

7.2.2.7 motorstatusmessage

```
void Motor::motorstatusmessage (
            const QString & message )  [signal]
```

motorstatusmessage emitted when the status of the serial connection changes, with a string indicating the actual state.

Parameters

message	the message

7.2.2.8 sensordata()

```
bool Motor::sensordata ( )
```

sensordata returns the current PCBMotor optical encoder wheel sendor state

Returns

the current PCBMotor optical encoder wheel sendor state

7.2.2.9 showStatusMessage

```
void Motor::showStatusMessage (
            const QString & message )  [slot]
```

showStatusMessage fills the label in the GUI with text

Parameters

message	the text to be shown in the label

7.2.2.10 stop

```
void Motor::stop (
            bool stop )  [slot]
```

stop Tries to stop movenents if possible

Parameters

stop	Input: True if movents shall be stopped if possible

7.2.2.11 write

```
void Motor::write (
            const QByteArray & data )  [slot]
```

write writes to the serialport

Parameters

data	data to be written to the serial port

7.2.3 Member Data Documentation

7.2.3.1 hometimer

```
QTimer Motor::hometimer  [private]
```

Used to iterate through the steps of 'go to the starting position' - but in an inaccurate way.

See also

command_home()

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

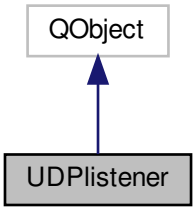
- motor.h
- motor.cpp

7.3 UDPlistener Class Reference

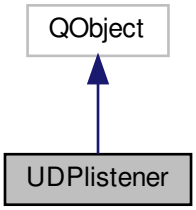
The UDPlistener class is used to control dinspect with UDP packages.

```
#include <udplistener.h>
```

Inheritance diagram for UDPlistener:



Collaboration diagram for UDPlistener:



Public Slots

- void bind ()

  bind Binds to a new UDP port

Signals

- void changeWPangles (QVector< double > angles)

  changeWPangles emitted when received message contains command to set waveplate characterization angles ('H')
- void changeoffsetusage (bool useoffset)

  changeoffsetusage emitted when massage to change usage of offset is received
- void Move (QString controller, double HWPang, double QWPang)

  Move emitted when massage to move the waveplates in a cage to certain angles is received.
- void MoveHV (QString controller)

  MoveHV emitted when massage to move cage to H/V basis is received.
- void MovePM (QString controller)

  MovePM emitted when massage to move cage to P/M basis is received.
- void MoveLR (QString controller)

  MoveLR emitted when massage to move cage to R/L basis is received.

Public Member Functions

- UDPlistener (QSettings *settings, QObject *parent=0)

  UDPlistener listen for commands on a UDP port and execute them.

Private Slots

- void processPendingDatagrams ()

  processPendingDatagrams reads data from the UDP socket
- void processCommands (QString msg)

  processCommands extracts commands out of received data and executes them

Private Attributes

- QSettings * settings

  configuration
- QUdpSocket socket

  the UDP socket
- uint port

  port the listener listens to
- bool alreadybound

  true if the listener is already listening to a port

7.3.1 Detailed Description

The UDPlistener class is used to control dinspect with UDP packages.

Bug There are no known bugs

7.3.2 Constructor & Destructor Documentation

7.3.2.1 UDPlistener()

```
UDPlistener::UDPlistener (
            QSettings * settings,
            QObject * parent = 0 )  [explicit]
```

UDPlistener listen for commands on a UDP port and execute them.

Parameters

settings	a pointer to a qsettings instance, used to get the port to bind to and known commands

UDPlistener opens a UDP socket and binds to a port specified in qsettings. Incoming packages are analysed to check if they contain known commands. If they do, these commands are executed.

The commands can be changed in the dinspect settings dialog, but the standard ones are:

- Move(QString, QString)
- Move(QString, double, double)

7.3.3 Member Function Documentation

7.3.3.1 bind

```
void UDPlistener::bind ( )  [slot]
```

bind Binds to a new UDP port

This function checks whether the UDP socket is already bound to a a specific port. If so, it closes this connection and binds to the new port. If not, it binds to the port right away.

7.3.3.2 changeoffsetusage

```
void UDPlistener::changeoffsetusage (
            bool useoffset )  [signal]
```

changeoffsetusage emitted when massage to change usage of offset is received

Parameters

useoffset	true if waveplate offset is to be used

7.3.3.3 changeWPangles

```
void UDPlistener::changeWPangles (
            QVector< double > angles )  [signal]
```

changeWPangles emitted when received message contains command to set waveplate characterization angles ('H')

Parameters

angles	vector containing all angles. Ordering: HWP0,HWP1,...,HWPn,QWP0,QWP1,...,QWPn

7.3.3.4 Move

```
void UDPlistener::Move (
            QString controller,
            double HWPang,
            double QWPang )  [signal]
```

Move emitted when massage to move the waveplates in a cage to certain angles is received.

Parameters

controller	either colorcode of cage or 'all'
HWPang	angle of the HWP in degree
QWPang	angle of the QWP in degree

7.3.3.5 MoveHV

```
void UDPlistener::MoveHV (
            QString controller )  [signal]
```

MoveHV emitted when massage to move cage to H/V basis is received.

Parameters

controller	either colorcode of stage, or 'all'

7.3.3.6 MoveLR

```
void UDPlistener::MoveLR (
```

```
            QString controller )  [signal]
```

MoveLR emitted when massage to move cage to R/L basis is received.

Parameters

controller	either colorcode of stage, or 'all'

7.3.3.7 MovePM

```
void UDPlistener::MovePM (
            QString controller )  [signal]
```

MovePM emitted when massage to move cage to P/M basis is received.

Parameters

controller	either colorcode of stage, or 'all'

7.3.3.8 processCommands

```
void UDPlistener::processCommands (
            QString msg )  [private], [slot]
```

processCommands extracts commands out of received data and executes them

Parameters

msg	input: the received message

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

- udplistener.h
- udplistener.cpp

Chapter 8

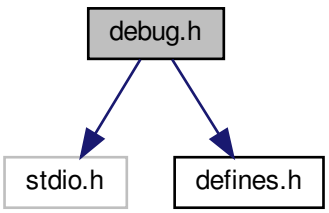
File Documentation

8.1 debug.h File Reference

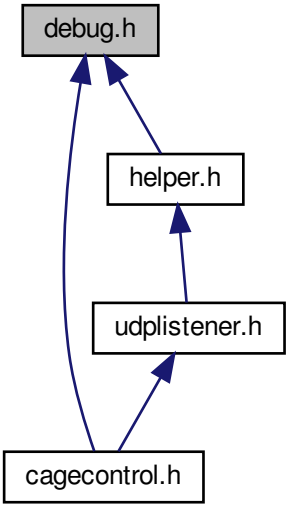
contains debug macros

```
#include "stdio.h"
#include "defines.h"
```

Include dependency graph for debug.h:



This graph shows which files directly or indirectly include this file:



8.1.1 Detailed Description

contains debug macros

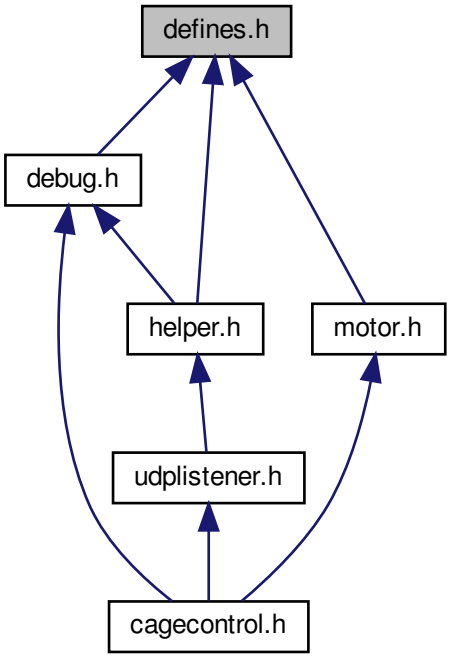
Bug Printing to console does not work on Windows. Workaround: Redirect stderr to stdout and redirect stdout to a file.

This file defines macros to style and simplify output to console.

8.2 defines.h File Reference

Various compile-time definitions.

This graph shows which files directly or indirectly include this file:



Macros

- #define **DEBUGSPECTROMETERCONFIG** FALSE
- #define UNUSED(expr) do { (void)(expr); } while (0)
- #define DEBUG 1
- #define DEBUGERROR 1
- #define DEBUGWARNING 1
- #define DEBUGINFO 1
- #define EPS 0.0000001
- #define PI 3.14159265358979323846
- #define DEGTORAD PI/180
- #define RADTODEG 180/PI

8.2.1 Detailed Description

Various compile-time definitions.

Bug There are no known bugs.

Contains definitions of various kind - mathematical, version constants, debug-variables, ...

8.2.2 Macro Definition Documentation

8.2.2.1 DEBUG

```
#define DEBUG 1
```

Enables the execution of various debug-paths used during development.
default: FALSE.

8.2.2.2 DEBUGERROR

```
#define DEBUGERROR 1
```

If set to TRUE, enables the execution of the DEBUG_ERROR() makro which is used to write error messages (critical) to stdout.
default: true

8.2.2.3 DEBUGINFO

```
#define DEBUGINFO 1
```

If set to TRUE, enables the execution of the DEBUG_INFO() makro which is used to write usefull information to stdout.
default: true

8.2.2.4 DEBUGWARNING

```
#define DEBUGWARNING 1
```

If set to TRUE, enables the execution of the DEBUG_WARNING() makro which is used to write warnings about unexpected behaviour to stdout.
default: true

8.2.2.5 DEGTORAD

```
#define DEGTORAD PI/180
```

Conversion factor from degree to radians. PI/180

8.2.2.6 EPS

```
#define EPS 0.0000001
```

'epsilon' used to check floatingpoint variables in if-conditions.
default: 0.0000001

8.2.2.7 PI

```
#define PI 3.14159265358979323846
```

Pi.
default: 3.14159265358979323846

8.2.2.8 RADTODEG

```
#define RADTODEG 180/PI
```

Conversion factor from radians to degree. 180/PI

8.2.2.9 UNUSED

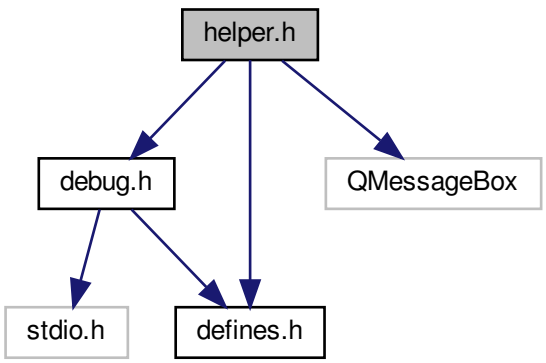
```
#define UNUSED(
              expr ) do { (void)(expr); } while (0)
```

Use UNUSED(var) in function f(...,type var,...) to silence compiler warnings about unused parameter var.

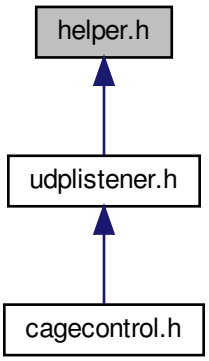
8.3 helper.h File Reference

```
#include "debug.h"
#include "defines.h"
#include <QMessageBox>
```

Include dependency graph for helper.h:



This graph shows which files directly or indirectly include this file:



Namespaces

- helper

  contains small functions to display messages

Functions

- void helper::message (QString msg)

  message displays a message box
- void helper::error (QString msg)

  error displays an error-messagebox and writes a debug_error message to stdout
- void helper::warning (QString msg)

  warning displays warning-messagebox and writes a debug_warning message to stdout
- void helper::info (QString msg)

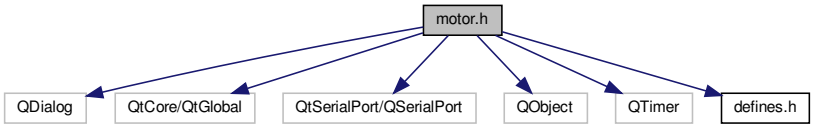
  info displays an info-messagebox and writes a debug_info message to stdout

8.4 motor.h File Reference

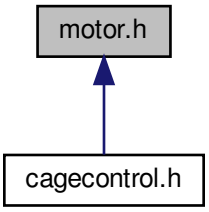
```
#include <QDialog>
#include <QtCore/QtGlobal>
#include <QtSerialPort/QSerialPort>
#include <QObject>
#include <QTimer>
```

```
#include "defines.h"
```

Include dependency graph for motor.h:



This graph shows which files directly or indirectly include this file:



Classes

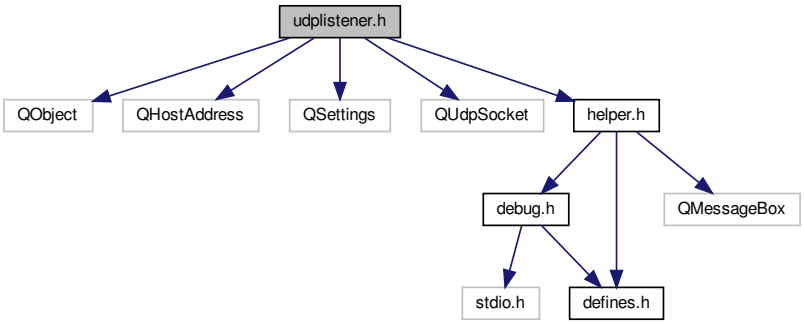
- class Motor

  The Motor class operates the PCB-motor.

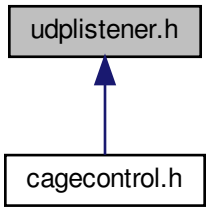
8.5 udplistener.h File Reference

```
#include <QObject>
#include <QHostAddress>
#include <QSettings>
#include <QUdpSocket>
#include "helper.h"
```

Include dependency graph for udplistener.h:



This graph shows which files directly or indirectly include this file:



Classes

- class UDPlistener

  The UDPlistener class is used to control dinspect with UDP packages.

8.6 version.h File Reference

This file contains information about the code version.

Macros

- #define VERSION_GIT "v0.1-9-g0c78d25"

  The git commit description.
- #define VERSION_GIT_DATE 201812181743

  The date of the git commit.
- #define VERSION_BUILD_DATE 201812181755

  The builddate.

8.6.1 Detailed Description

This file contains information about the code version.

Author

Peter Schiansky

Bug There are no known bugs.

The definitions in this file are used to fill the about-dialog with information about the code: The git commit description, the commit date and the builddate.

Index