# 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.

**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**

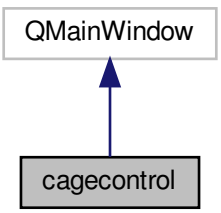
| | |
|---|---|
| *msg* | the message to be displayed |

# Chapter 7

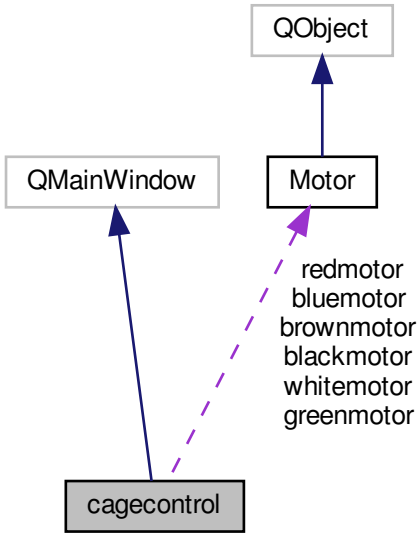
# Class Documentation

## 7.1 cagecontrol Class Reference

Inheritance diagram for cagecontrol:



Collaboration diagram for cagecontrol:




## Public Member Functions

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

## Private Slots

- void **updatesettings** (double d)
- void **updateUI** ()

## Private Member Functions

- void **setupUI** (QGridLayout *layout)
- void **openmotors** ()
- void LoadConfig ()

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

  *SaveConfig stores config to a file.*
- void **motorGB** (QGroupBox *gb, QString id)
- void **initconnections** ()
- void **movemotor** (QString motor, double HWPang, double QWPang)
- void **moveredHV** ()
- void **moveredPM** ()
- void **moveredANG** ()
- void **movebrownHV** ()
- void **movebrownPM** ()
- void **movebrownANG** ()

- void **movegreenHV** ()
- void **movegreenPM** ()
- void **movegreenANG** ()
- void **moveblueHV** ()
- void **movebluePM** ()
- void **moveblueANG** ()
- void **movewhiteHV** ()
- void **movewhitePM** ()
- void **movewhiteANG** ()
- void **moveblackHV** ()
- void **moveblackPM** ()
- void **moveblackANG** ()

### Private Attributes

- QSettings * **settings**
- QTabWidget * **tabs**
- QWidget * **settingstab**
- QWidget * **motorstab**
- QVector< QString > comports

  *Vector containing available serial ports names ports.*
- Motor * **redmotor**
- Motor * **brownmotor**
- Motor * **greenmotor**
- Motor * **bluemotor**
- Motor * **whitemotor**
- Motor * **blackmotor**
- QVector< Motor * > **motors**
- QVector< QString > **motorName**
- QVector< QDoubleSpinBox > **HWP0sp**
- QVector< QDoubleSpinBox > **QWP0sp**
- QVector< int > **HWPmnum**
- QVector< int > **QWPmnum**
- QVector< double > **HWP0**
- QVector< double > **QWP0**
- QVector< double > **HWPcust**
- QVector< double > **QWPcust**

## 7.1.1 Member Function Documentation

### 7.1.1.1 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.

#### 7.1.1.2 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.

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

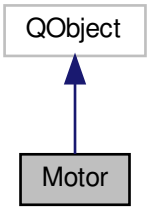
- cagecontrol.h
- cagecontrol.cpp

## 7.2 Motor Class Reference

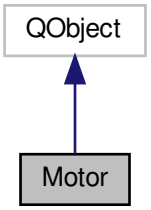
The Motor class operates the PCB-motor.

```
#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 boththimer

  *Used to iterate through the steps of moving two motors of one controller.*
- int **movebothstep**
- bool **serialconnectionok**
- uint16_t **motor1steps**
- uint16_t **motor2steps**

## 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

# Chapter 8

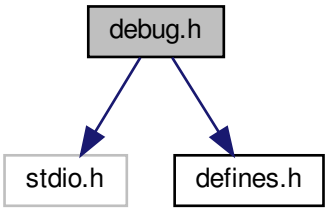
# File Documentation

## 8.1 debug.h File Reference

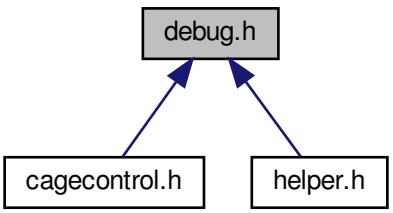
contains debug macros

```
#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

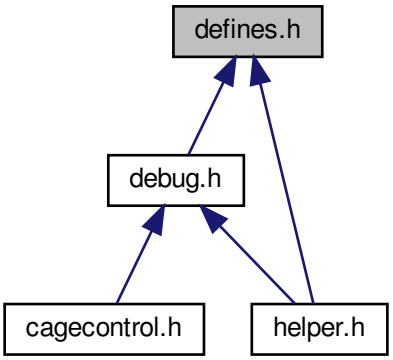
**Bug** Printing to console does not work on Windows. Workaround: Redirect stderr to stdout and redirect stdout to a file.

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 DEBUG true
- #define DEBUGERROR true
- #define DEBUGWARNING true
- #define DEBUGINFO true
- #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 true
```

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

#### 8.2.2.2 DEBUGERROR

```
#define DEBUGERROR true
```

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 true
```

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 true
```

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

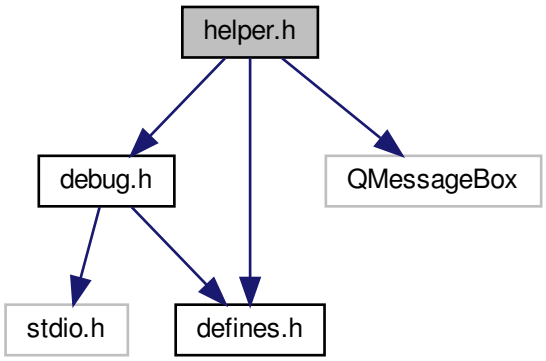
```
#define RADTODEG 180/PI
```

Conversion factor from radians to degree. 180/PI

## 8.3 helper.h File Reference

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

Include dependency graph for helper.h:




### 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)

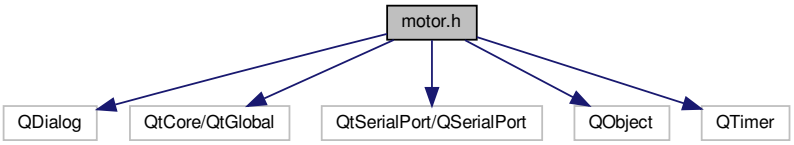
  *warning displays warning-messagebox and writes a debug_warning message to stdout*
- void helper::info (QString msg)

  *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 dependency graph for motor.h:




### Classes

- class Motor

  *The Motor class operates the PCB-motor.*

## 8.5 version.h File Reference

This file contains information about the code version.

### Macros

- #define VERSION_GIT ""

  *The git commit description.*
- #define VERSION_GIT_DATE 201812121130

  *The date of the git commit.*
- #define VERSION_BUILD_DATE 201812121153

  *The builddate.*

### 8.5.1 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