

## Sistema de Balanças Automotivo

Generated by Doxygen 1.9.2

<b>1 File Index</b>	<b>1</b>
1.1 File List	1
<b>2 File Documentation</b>	<b>3</b>
2.1 main.cpp File Reference	3
2.1.1 Function Documentation	4
2.1.1.1 loop()	4
2.1.1.2 mediaMoveI()	4
2.1.1.3 printLat()	4
2.1.1.4 printLong()	4
2.1.1.5 printScales()	4
2.1.1.6 printTotal()	5
2.1.1.7 processData()	5
2.1.1.8 readButtons()	5
2.1.1.9 readData()	5
2.1.1.10 setup()	5
2.1.1.11 tara()	5
2.1.1.12 TaskLeitura()	6
2.1.1.13 TaskUpdateLCD()	6
2.1.2 Variable Documentation	6
2.1.2.1 threadSemaphore	6
2.2 main.h File Reference	6
2.2.1 Macro Definition Documentation	8
2.2.1.1 bitshift	8
2.2.1.2 BUFFER_SIZE	8
2.2.1.3 FD_PORT	8
2.2.1.4 FE_PORT	8
2.2.1.5 FUNC_PIN	8
2.2.1.6 T_PIN	8
2.2.1.7 TD_PORT	9
2.2.1.8 TE_PORT	9
2.2.1.9 UPDATE_LCD_HZ	9
2.2.2 Function Documentation	9
2.2.2.1 lcd()	9
2.2.2.2 mediaMoveI()	9
2.2.2.3 printLat()	9
2.2.2.4 printLong()	10
2.2.2.5 printScales()	10
2.2.2.6 printTotal()	10
2.2.2.7 processData()	10
2.2.2.8 readButtons()	10
2.2.2.9 readData()	10

---

2.2.2.10 tara()	11
2.2.3 Variable Documentation	11
2.2.3.1 calibrationFactorDd	11
2.2.3.2 calibrationFactorDe	11
2.2.3.3 calibrationFactorTd	11
2.2.3.4 calibrationFactorTe	11
2.2.3.5 d4	11
2.2.3.6 d5	12
2.2.3.7 d6	12
2.2.3.8 d7	12
2.2.3.9 dataDd	12
2.2.3.10 dataDe	12
2.2.3.11 dataTd	12
2.2.3.12 dataTe	12
2.2.3.13 dd	13
2.2.3.14 de	13
2.2.3.15 en	13
2.2.3.16 funcState	13
2.2.3.17 rs	13
2.2.3.18 state	13
2.2.3.19 taraDd	13
2.2.3.20 taraDe	14
2.2.3.21 taraTd	14
2.2.3.22 taraTe	14
2.2.3.23 td	14
2.2.3.24 te	14
2.2.3.25 total	14
2.2.3.26 tState	14
2.3 main.h	15
<b>Index</b>	<b>17</b>

# Chapter 1

## File Index

### 1.1 File List

Here is a list of all files with brief descriptions:

<b>main.cpp</b>	3
<b>main.h</b>	6

## Chapter 2

# File Documentation

### 2.1 main.cpp File Reference

```
#include <Arduino.h>
#include "main.h"
#include <Arduino_FreeRTOS.h>
#include <semphr.h>
#include <queue.h>
#include <task.h>
```

#### Functions

- void **TaskLeitura** (void \*pvParameters)  
*include FreeRTOS librarys*
- void **TaskUpdateLCD** (void \*pvParameters)
- void **setup** ()  
*mutex que vai controlar porta serial*
- void **loop** ()
- void **readButtons** ()  
*read state of each button*
- void **printScales** ()  
*print data in LCD about individual scales data*
- void **printLong** ()  
*print data in LCD about longitudinal mass distribution*
- void **printLat** ()  
*print data in LCD about lateral mass distribution*
- void **printTotal** ()  
*print data in LCD about total mass of the car*
- void **readData** ()  
*read analog data and store its in an vector*
- void **tara** ()
- void **processData** ()
- int32\_t **mediaMoveI** (int32\_t \*array)  
*calculate media moveI of an vector*

## Variables

- SemaphoreHandle\_t **threadSemaphore**

## 2.1.1 Function Documentation

### 2.1.1.1 loop()

```
void loop ( )
```

### 2.1.1.2 mediaMove()

```
int32_t mediaMove (
    int32_t * array )
```

calculate media move of an vector

### 2.1.1.3 printLat()

```
void printLat ( )
```

print data in LCD about lateral mass distribution

### 2.1.1.4 printLong()

```
void printLong ( )
```

print data in LCD about longitudinal mass distribution

### 2.1.1.5 printScales()

```
void printScales ( )
```

print data in LCD about individual scales data

functions

#### 2.1.1.6 printTotal()

```
void printTotal ( )
```

print data in LCD about total mass of the car

#### 2.1.1.7 processData()

```
void processData ( )
```

process data, calculating media moveI, removing tara, and converting to kg

#### 2.1.1.8 readButtons()

```
void readButtons ( )
```

read state of each button

#### 2.1.1.9 readData()

```
void readData ( )
```

read analog data and store its in an vector

#### 2.1.1.10 setup()

```
void setup ( )
```

mutex que vai controlar porta serial

initialize serial communication at 9600 bits per second:

read EEPROM data

cria a mutex

Now set up two tasks to run independently.

#### 2.1.1.11 tara()

```
void tara ( )
```

get the current analog data and store its to EEPROM this value will be subtract from de current data that is being redde

#### 2.1.1.12 TaskLeitura()

```
void TaskLeitura (
    void * pvParameters )
```

include FreeRTOS librarys

define two tasks for read and print data

#### 2.1.1.13 TaskUpdateLCD()

```
void TaskUpdateLCD (
    void * pvParameters )
```

### 2.1.2 Variable Documentation

#### 2.1.2.1 threadSemaphore

```
SemaphoreHandle_t threadSemaphore
```

## 2.2 main.h File Reference

```
#include <LiquidCrystal.h>
#include <EEPROM.h>
```

### Macros

- #define **FE\_PORT** 0  
*defines analog inputs pin*
- #define **FD\_PORT** 1
- #define **TE\_PORT** 2
- #define **TD\_PORT** 3
- #define **FUNC\_PIN** 2  
*defines digital inputs pin*
- #define **T\_PIN** 3
- #define **BUFFER\_SIZE** 32  
*defines constants of buffer and media movel calculation*
- #define **bitshift** 5
- #define **UPDATE\_LCD\_HZ** 10



## Functions

- LiquidCrystal **lcd** ( **rs**, **en**, **d4**, **d5**, **d6**, **d7**)
- void **printScales** ()  
*functions*
- void **printLong** ()  
*print data in LCD about longitudinal mass distribution*
- void **printLat** ()  
*print data in LCD about lateral mass distribution*
- void **printTotal** ()  
*print data in LCD about total mass of the car*
- void **tara** ()
- void **readButtons** ()  
*read state of each button*
- void **readData** ()  
*read analog data and store its in an vector*
- void **processData** ()
- int32\_t **mediaMoveI** (int32\_t \*array)  
*calculate media moveI of an vector*

## Variables

- const int **rs** = 13
- const int **en** = 12
- const int **d4** = 11
- const int **d5** = 10
- const int **d6** = 9
- const int **d7** = 8
- const double **calibrationFactorDe** = 0.48481  
*initialize calibration data that converts Voltage (0-1023) to Kg*
- const double **calibrationFactorDd** = 0.48481
- const double **calibrationFactorTe** = 0.48481
- const double **calibrationFactorTd** = 0.48481
- byte **state** = 0  
*state of LCD (witch function is beeing displayed)*
- boolean **funcState** = false  
*state of each button*
- boolean **tState** = false
- int32\_t **dataDe** [ **BUFFER\_SIZE**]  
*data buffers*
- int32\_t **dataDd** [ **BUFFER\_SIZE**]
- int32\_t **dataTe** [ **BUFFER\_SIZE**]
- int32\_t **dataTd** [ **BUFFER\_SIZE**]
- int32\_t **taraDe** = 0  
*data constants*
- int32\_t **taraDd** = 0
- int32\_t **taraTe** = 0
- int32\_t **taraTd** = 0
- double **total** = 0
- double **de** = 0
- double **dd** = 0
- double **te** = 0
- double **td** = 0

## 2.2.1 Macro Definition Documentation

### 2.2.1.1 bitshift

```
#define bitshift 5
```

### 2.2.1.2 BUFFER\_SIZE

```
#define BUFFER_SIZE 32
```

defines constants of buffer and media move calculation

### 2.2.1.3 FD\_PORT

```
#define FD_PORT 1
```

### 2.2.1.4 FE\_PORT

```
#define FE_PORT 0
```

defines analog inputs pin

### 2.2.1.5 FUNC\_PIN

```
#define FUNC_PIN 2
```

defines digital inputs pin

### 2.2.1.6 T\_PIN

```
#define T_PIN 3
```

### 2.2.1.7 TD\_PORT

```
#define TD_PORT 3
```

### 2.2.1.8 TE\_PORT

```
#define TE_PORT 2
```

### 2.2.1.9 UPDATE\_LCD\_HZ

```
#define UPDATE_LCD_HZ 10
```

## 2.2.2 Function Documentation

### 2.2.2.1 lcd()

```
LiquidCrystal lcd (  
    rs ,  
    en ,  
    d4 ,  
    d5 ,  
    d6 ,  
    d7 )
```

### 2.2.2.2 mediaMove()

```
int32_t mediaMove (  
    int32_t * array )
```

calculate media move of an vector

### 2.2.2.3 printLat()

```
void printLat ( )
```

print data in LCD about lateral mass distribution

#### 2.2.2.4 printLong()

```
void printLong ( )
```

print data in LCD about longitudinal mass distribution

#### 2.2.2.5 printScales()

```
void printScales ( )
```

functions

functions

#### 2.2.2.6 printTotal()

```
void printTotal ( )
```

print data in LCD about total mass of the car

#### 2.2.2.7 processData()

```
void processData ( )
```

process data, calculating media moveI, removing tara, and converting to kg

#### 2.2.2.8 readButtons()

```
void readButtons ( )
```

read state of each button

#### 2.2.2.9 readData()

```
void readData ( )
```

read analog data and store its in an vector

### 2.2.2.10 tara()

```
void tara ( )
```

get the current analog data and store its to EEPROM this value will be subtract from de current data that is being reddened

## 2.2.3 Variable Documentation

### 2.2.3.1 calibrationFactorDd

```
const double calibrationFactorDd = 0.48481
```

### 2.2.3.2 calibrationFactorDe

```
const double calibrationFactorDe = 0.48481
```

initialize calibration data that converts Voltage (0-1023) to Kg

### 2.2.3.3 calibrationFactorTd

```
const double calibrationFactorTd = 0.48481
```

### 2.2.3.4 calibrationFactorTe

```
const double calibrationFactorTe = 0.48481
```

### 2.2.3.5 d4

```
const int d4 = 11
```

#### 2.2.3.6 d5

```
const int d5 = 10
```

#### 2.2.3.7 d6

```
const int d6 = 9
```

#### 2.2.3.8 d7

```
const int d7 = 8
```

#### 2.2.3.9 dataDd

```
int32_t dataDd[ BUFFER_SIZE]
```

#### 2.2.3.10 dataDe

```
int32_t dataDe[ BUFFER_SIZE]
```

data buffers

#### 2.2.3.11 dataTd

```
int32_t dataTd[ BUFFER_SIZE]
```

#### 2.2.3.12 dataTe

```
int32_t dataTe[ BUFFER_SIZE]
```

### 2.2.3.13 dd

```
double dd = 0
```

### 2.2.3.14 de

```
double de = 0
```

### 2.2.3.15 en

```
const int en = 12
```

### 2.2.3.16 funcState

```
boolean funcState = false
```

state of each button

### 2.2.3.17 rs

```
const int rs = 13
```

### 2.2.3.18 state

```
byte state = 0
```

state of LCD (witch function is beeing displayed)

### 2.2.3.19 taraDd

```
int32_t taraDd = 0
```

**2.2.3.20 taraDe**

```
int32_t taraDe = 0
```

data constants

**2.2.3.21 taraTd**

```
int32_t taraTd = 0
```

**2.2.3.22 taraTe**

```
int32_t taraTe = 0
```

**2.2.3.23 td**

```
double td = 0
```

**2.2.3.24 te**

```
double te = 0
```

**2.2.3.25 total**

```
double total = 0
```

**2.2.3.26 tState**

```
boolean tState = false
```



## 2.3 main.h

**Go to the documentation of this file.**

```
1 //
2 // Created by eugen on 26/08/2021.
3 //
4
5 // include the library code:
6 #include <LiquidCrystal.h>
7 #include <EEPROM.h>
8
9 #define FE_PORT 0
10 #define FD_PORT 1
11 #define TE_PORT 2
12 #define TD_PORT 3
13
14 #define FUNC_PIN 2
15 #define T_PIN 3
16
17 #define BUFFER_SIZE 32
18 #define bitshift 5
19 #define UPDATE_LCD_HZ 10
20
21 // initialize the LCD by associating any needed LCD interface pin
22 // with the arduino pin number it is connected to
23 const int rs = 13, en = 12, d4 = 11, d5 = 10, d6 = 9, d7 = 8;
24 LiquidCrystal lcd(rs, en, d4, d5, d6, d7);
25
26 const double calibrationFactorDe = 0.48481;
27 const double calibrationFactorDd = 0.48481;
28 const double calibrationFactorTe = 0.48481;
29 const double calibrationFactorTd = 0.48481;
30
31 byte state = 0;
32
33 boolean funcState = false;
34 boolean tState = false;
35
36 int32_t dataDe[BUFFER_SIZE];
37 int32_t dataDd[BUFFER_SIZE];
38 int32_t dataTe[BUFFER_SIZE];
39 int32_t dataTd[BUFFER_SIZE];
40
41 int32_t taraDe = 0;
42 int32_t taraDd = 0;
43 int32_t taraTe = 0;
44 int32_t taraTd = 0;
45
46 double total = 0;
47 double de = 0;
48 double dd = 0;
49 double te = 0;
50 double td = 0;
51
52 void printScales();
53 void printLong();
54 void printLat();
55 void printTotal();
56 void tara();
57
58 void readButtons();
59 void readData();
60 void processData();
61 int32_t mediaMove1(int32_t *array);
```



# Index

- bitshift
  - main.h, 8
- BUFFER\_SIZE
  - main.h, 8
- calibrationFactorDd
  - main.h, 11
- calibrationFactorDe
  - main.h, 11
- calibrationFactorTd
  - main.h, 11
- calibrationFactorTe
  - main.h, 11
- d4
  - main.h, 11
- d5
  - main.h, 11
- d6
  - main.h, 12
- d7
  - main.h, 12
- dataDd
  - main.h, 12
- dataDe
  - main.h, 12
- dataTd
  - main.h, 12
- dataTe
  - main.h, 12
- dd
  - main.h, 12
- de
  - main.h, 13
- en
  - main.h, 13
- FD\_PORT
  - main.h, 8
- FE\_PORT
  - main.h, 8
- FUNC\_PIN
  - main.h, 8
- funcState
  - main.h, 13
- lcd
  - main.h, 9
- loop
  - main.cpp, 4

- main.cpp, 3
  - loop, 4
  - mediaMoveI, 4
  - printLat, 4
  - printLong, 4
  - printScales, 4
  - printTotal, 4
  - processData, 5
  - readButtons, 5
  - readData, 5
  - setup, 5
  - tara, 5
  - TaskLeitura, 5
  - TaskUpdateLCD, 6
  - threadSemaphore, 6
- main.h, 6
  - bitshift, 8
  - BUFFER\_SIZE, 8
  - calibrationFactorDd, 11
  - calibrationFactorDe, 11
  - calibrationFactorTd, 11
  - calibrationFactorTe, 11
  - d4, 11
  - d5, 11
  - d6, 12
  - d7, 12
  - dataDd, 12
  - dataDe, 12
  - dataTd, 12
  - dataTe, 12
  - dd, 12
  - de, 13
  - en, 13
  - FD\_PORT, 8
  - FE\_PORT, 8
  - FUNC\_PIN, 8
  - funcState, 13
  - lcd, 9
  - mediaMoveI, 9
  - printLat, 9
  - printLong, 9
  - printScales, 10
  - printTotal, 10
  - processData, 10
  - readButtons, 10
  - readData, 10
  - rs, 13
  - state, 13
  - T\_PIN, 8

- tara, 10
- taraDd, 13
- taraDe, 13
- taraTd, 14
- taraTe, 14
- td, 14
- TD\_PORT, 8
- te, 14
- TE\_PORT, 9
- total, 14
- tState, 14
- UPDATE\_LCD\_HZ, 9
- mediaMoveI
  - main.cpp, 4
  - main.h, 9
- printLat
  - main.cpp, 4
  - main.h, 9
- printLong
  - main.cpp, 4
  - main.h, 9
- printScales
  - main.cpp, 4
  - main.h, 10
- printTotal
  - main.cpp, 4
  - main.h, 10
- processData
  - main.cpp, 5
  - main.h, 10
- readButtons
  - main.cpp, 5
  - main.h, 10
- readData
  - main.cpp, 5
  - main.h, 10
- rs
  - main.h, 13
- setup
  - main.cpp, 5
- state
  - main.h, 13
- T\_PIN
  - main.h, 8
- tara
  - main.cpp, 5
  - main.h, 10
- taraDd
  - main.h, 13
- taraDe
  - main.h, 13
- taraTd
  - main.h, 14
- taraTe
  - main.h, 14
- TaskLeitura
  - main.cpp, 5
- TaskUpdateLCD
  - main.cpp, 6
- td
  - main.h, 14
- TD\_PORT
  - main.h, 8
- te
  - main.h, 14
- TE\_PORT
  - main.h, 9
- threadSemaphore
  - main.cpp, 6
- total
  - main.h, 14
- tState
  - main.h, 14
- UPDATE\_LCD\_HZ
  - main.h, 9