Firmware BRDQBRQBC = HFC2 and BRDQBRQBC = HFS1 documentation - qbcontrol main board

Generated by Doxygen 1.8.13

Contents

1 Firmware **1**

2 Data Structure Index **3**

2.1 Data Structures 3

3 File Index **5**

3.1 File List 5

4 Data Structure Documentation **7**

4.1 st_data Struct Reference 7

4.2 st_meas Struct Reference 7

4.3 st_mem Struct Reference 8

4.4 st_ref Struct Reference 8

5 File Documentation **9**

5.1 command_processing.c File Reference 9

5.1.1 Detailed Description 10

5.1.2 Function Documentation 10

5.1.2.1 cmd_get_measurements() 11

5.1.2.2 memInit() 11

5.1.2.3 memRecall() 11

5.1.2.4 memRestore() 11

5.1.2.5 memStore() 11

5.2 command_processing.h File Reference 12

5.2.1 Detailed Description 13

5.2.2 Function Documentation . . . 13
5.2.2.1 cmd_get_measurements() . . . 13
5.2.2.2 memInit() . . . 14
5.2.2.3 memRecall() . . . 14
5.2.2.4 memRestore() . . . 14
5.2.2.5 memStore() . . . 14
5.3 commands.h File Reference . . . 14
5.3.1 Detailed Description . . . 15
5.3.2 Enumeration Type Documentation . . . 16
5.3.2.1 qbmove_command . . . 16
5.3.2.2 qbmove_control_mode . . . 17
5.3.2.3 qbmove_input_mode . . . 17
5.3.2.4 qbmove_parameter . . . 17
5.4 globals.c File Reference . . . 18
5.4.1 Detailed Description . . . 19
5.5 globals.h File Reference . . . 20
5.5.1 Detailed Description . . . 22
5.6 interruptions.c File Reference . . . 22
5.6.1 Detailed Description . . . 23
5.6.2 Variable Documentation . . . 24
5.6.2.1 pwm_preload_values . . . 24
5.7 interruptions.h File Reference . . . 25
5.7.1 Detailed Description . . . 26
5.8 main.c File Reference . . . 26
5.8.1 Detailed Description . . . 27
5.9 utils.h File Reference . . . 27
5.9.1 Detailed Description . . . 28

Index **31**

Chapter 1

Firmware

This is the firmware of the qbcontrol main board for CUFF and Stretch Pro haptic feedback devices.

Version

1.0

This is the firmware of the CUFF and Stretch Pro haptic feedback devices. It can control two motors and read their encoders. Also can read and convert analog measurements connected to the PSoC microcontroller.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

st_data 7
st_meas 7
st_mem 8
st_ref 8

Chapter 3

File Index

3.1 File List

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

command_processing.c
Command processing functions . . . 9
command_processing.h
Command processing functions . . . 12
commands.h
Definitions for commands, parameters and packages . . . 14
device.h . . . **??**
globals.c
Global variables . . . 18
globals.h
Global definitions and macros are set in this file . . . 20
interruptions.c
Interruption functions are in this file . . . 22
interruptions.h
Interruptions header file . . . 25
main.c
Firmware main file . . . 26
utils.h
Definition of utility functions . . . 27

Chapter 4

Data Structure Documentation

4.1 st_data Struct Reference

Data Fields

- uint8 **buffer** [128]
- int16 **length**
- int16 **ind**
- uint8 **ready**

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

- **globals.h**

4.2 st_meas Struct Reference

Data Fields

- int32 **pos** [NUM_OF_SENSORS]
- int16 **curr** [NUM_OF_MOTORS]
- int8 **rot** [NUM_OF_SENSORS]
- int16 **vel** [NUM_OF_SENSORS]
- int16 **acc** [NUM_OF_SENSORS]

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

- **globals.h**

4.3 st_mem Struct Reference

Data Fields

- uint8 **flag**
- uint8 **id**
- int32 **k_p**
- int32 **k_i**
- int32 **k_d**
- int32 **k_p_c**
- int32 **k_i_c**
- int32 **k_d_c**
- int32 **k_p_dl**
- int32 **k_i_dl**
- int32 **k_d_dl**
- int32 **k_p_c_dl**
- int32 **k_i_c_dl**
- int32 **k_d_c_dl**
- int16 **current_limit**
- uint8 **activ**
- uint8 **input_mode**
- uint8 **control_mode**
- uint8 **res** [NUM_OF_SENSORS]
- int32 **m_off** [NUM_OF_SENSORS]
- float **m_mult** [NUM_OF_SENSORS]
- uint8 **pos_lim_flag**
- int32 **pos_lim_inf** [NUM_OF_MOTORS]
- int32 **pos_lim_sup** [NUM_OF_MOTORS]
- uint16 **max_stiffness**
- uint8 **baud_rate**
- uint8 **watchdog_period**
- int32 **max_step_neg**
- int32 **max_step_pos**
- uint8 **cuff_activation_flag**
- float **curr_prop_gain**
- int16 **curr_sat**
- int16 **curr_dead_zone**
- uint16 **power_tension**

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

- **globals.h**

4.4 st_ref Struct Reference

Data Fields

- int32 **pos** [NUM_OF_MOTORS]
- uint8 **onoff**

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

- **globals.h**

Chapter 5

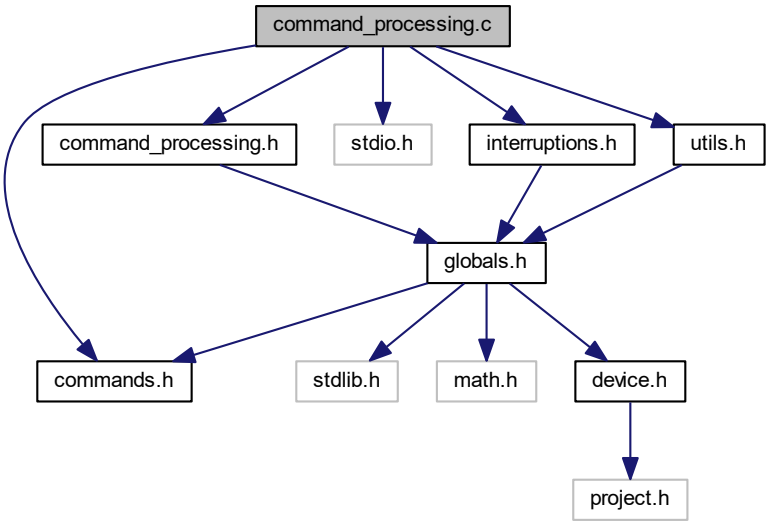
File Documentation

5.1 command_processing.c File Reference

Command processing functions.

```
#include <command_processing.h>
#include <stdio.h>
#include <interruptions.h>
#include <utils.h>
#include "commands.h"
```

Include dependency graph for command_processing.c:



Functions

- void **commProcess** ()

- void **drive_cuff** ()
- void **infoGet** (uint16 info_type)
- void **setZeros** ()
- void **get_param_list** (uint16 index)
- void **infoPrepare** (unsigned char ∗info_string)
- void **commWrite_old_id** (uint8 ∗packet_data, uint16 packet_lenght, uint8 old_id)
- void **commWrite** (uint8 ∗packet_data, const uint16 packet_lenght, uint8 next)
- void **sendAcknowledgment** (const uint8 value)
- uint8 **memStore** (int displacement)
- void **memRecall** (void)
- uint8 **memRestore** (void)
- uint8 **memInit** (void)
- void **cmd_get_measurements** ()
- void **cmd_get_inputs** ()
- void **cmd_get_currents** ()
- void **cmd_get_curr_and_meas** ()
- void **cmd_set_inputs** ()
- void **cmd_set_pos_stiff** ()
- void **cmd_get_velocities** ()
- void **cmd_activate** ()
- void **cmd_set_watchdog** ()
- void **cmd_get_activate** ()
- void **cmd_ping** ()
- void **cmd_store_params** ()
- void **cmd_set_baudrate** ()

Variables

- reg8 ∗ **EEPROM_ADDR** = (reg8 ∗) CYDEV_EE_BASE

5.1.1 Detailed Description

Command processing functions.

Date

October 01, 2017

Author

Centro "E.Piaggio"

Copyright

5.1.2 Function Documentation

5.1.2.1 cmd_get_measurements()

```
void cmd_get_measurements ( )
```

Bunch of functions used on request from UART communication

5.1.2.2 memInit()

```
uint8 memInit (
             void  )
```

This function initialize memory when eeprom is compromised.

5.1.2.3 memRecall()

```
void memRecall (
             void  )
```

This function loads user settings from the eeprom.

5.1.2.4 memRestore()

```
uint8 memRestore (
             void  )
```

This function loads default settings from the eeprom.

5.1.2.5 memStore()

```
uint8 memStore (
             int displacement )
```

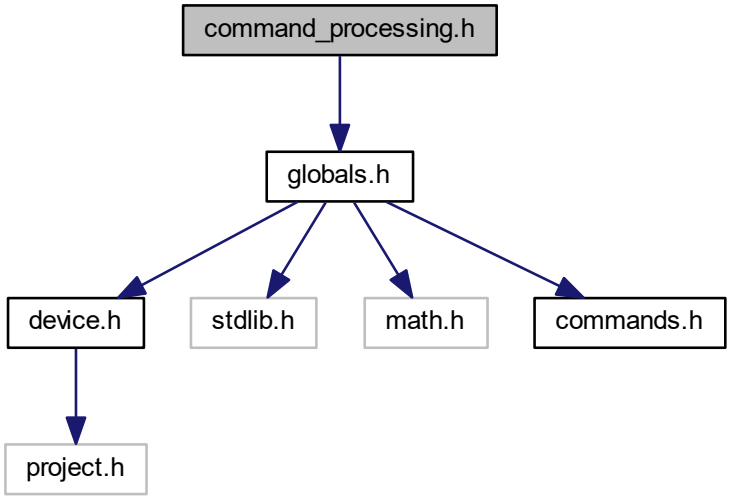
This function stores current memory settings on the eeprom with the specified displacement

5.2 command_processing.h File Reference

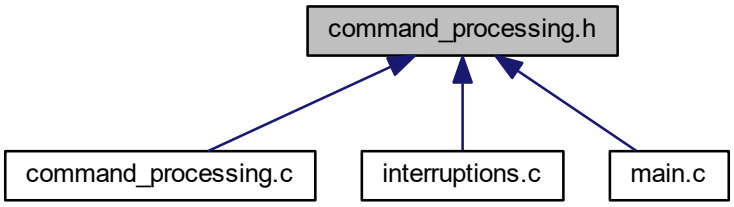
Command processing functions.

```
#include <globals.h>
```

Include dependency graph for command_processing.h:



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



Functions

- void **setZeros** (void)
- void **get_param_list** (uint16 index)
- void **infoPrepare** (unsigned char *)
- void **infoGet** (uint16)
- void **commProcess** ()

- void **drive_cuff** ()
- void **commWrite** (uint8 *, const uint16, uint8)
- void **commWrite_old_id** (uint8 *, const uint16, uint8)
- uint8 **memStore** (int)
- void **sendAcknowledgment** (const uint8)
- void **memRecall** (void)
- uint8 **memRestore** (void)
- uint8 **memInit** (void)
- void **cmd_get_measurements** ()
- void **cmd_get_inputs** ()
- void **cmd_get_currents** ()
- void **cmd_get_curr_and_meas** ()
- void **cmd_set_inputs** ()
- void **cmd_set_pos_stiff** ()
- void **cmd_get_velocities** ()
- void **cmd_activate** ()
- void **cmd_set_watchdog** ()
- void **cmd_get_activate** ()
- void **cmd_ping** ()
- void **cmd_store_params** ()
- void **cmd_set_baudrate** ()

5.2.1 Detailed Description

Command processing functions.

Date

October 01, 2017

Author

Centro "E.Piaggio"

Copyright

5.2.2 Function Documentation

5.2.2.1 cmd_get_measurements()

```
void cmd_get_measurements ( )
```

Bunch of functions used on request from UART communication

5.2.2.2 memInit()

```
uint8 memInit (
            void  )
```

This function initialize memory when eeprom is compromised.

5.2.2.3 memRecall()

```
void memRecall (
            void  )
```

This function loads user settings from the eeprom.

5.2.2.4 memRestore()

```
uint8 memRestore (
            void  )
```

This function loads default settings from the eeprom.

5.2.2.5 memStore()

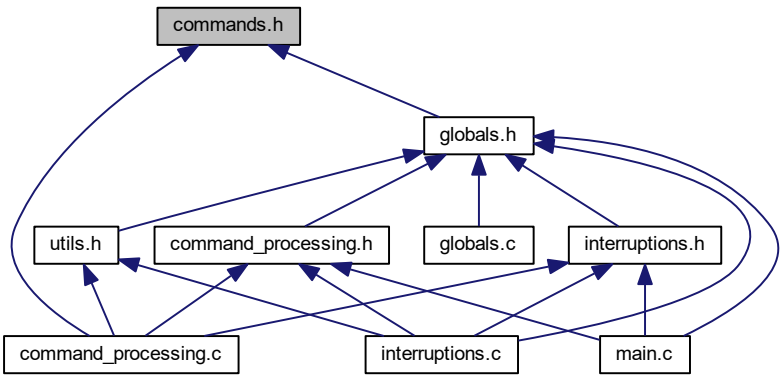
```
uint8 memStore (
            int displacement )
```

This function stores current memory settings on the eeprom with the specified displacement

5.3 commands.h File Reference

Definitions for commands, parameters and packages.

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



Enumerations

QB Move Commands

- enum **qbmove_command** {
  CMD_PING = 0, **CMD_SET_ZEROS** = 1, **CMD_STORE_PARAMS** = 3, **CMD_STORE_DEFAULT_P↩ ARAMS** = 4,
  CMD_RESTORE_PARAMS = 5, **CMD_GET_INFO** = 6, **CMD_SET_VALUE** = 7, **CMD_GET_VALUE** = 8,
  CMD_BOOTLOADER = 9, **CMD_INIT_MEM** = 10, **CMD_CALIBRATE** = 11, **CMD_GET_PARAM_LIST** = 12,
  CMD_HAND_CALIBRATE = 13, **CMD_ACTIVATE** = 128, **CMD_GET_ACTIVATE** = 129, **CMD_SET↩ _INPUTS** = 130,
  CMD_GET_INPUTS = 131, **CMD_GET_MEASUREMENTS** = 132, **CMD_GET_CURRENTS** = 133, **C↩ MD_GET_CURR_AND_MEAS** = 134,
  CMD_SET_POS_STIFF = 135, **CMD_GET_VELOCITIES** = 137, **CMD_GET_COUNTERS** = 138, **CM↩ D_GET_ACCEL** = 139,
  CMD_GET_CURR_DIFF = 140, **CMD_SET_CURR_DIFF** = 141, **CMD_SET_CUFF_INPUTS** = 142, **C↩ MD_SET_WATCHDOG** = 143,
  CMD_SET_BAUDRATE = 144 }

QB Move Parameters

- #define **PARAM_BYTE_SLOT** 50
- #define **PARAM_MENU_SLOT** 150
- #define **INFO_ALL** 0
- enum **qbmove_parameter** {
  PARAM_ID = 0, **PARAM_PID_CONTROL** = 1, **PARAM_STARTUP_ACTIVATION** = 2, **PARAM_INPU↩ T_MODE** = 3,
  PARAM_CONTROL_MODE = 4, **PARAM_MEASUREMENT_OFFSET** = 5, **PARAM_MEASUREMENT↩ _MULTIPLIER** = 6, **PARAM_POS_LIMIT_FLAG** = 7,
  PARAM_POS_LIMIT = 8, **PARAM_MAX_STEP_POS** = 9, **PARAM_MAX_STEP_NEG** = 10, **PARAM_↩ POS_RESOLUTION** = 11,
  PARAM_CURRENT_LIMIT = 12, **PARAM_PID_CURR_CONTROL** = 18, **PARAM_CURR_PROP_GAIN** = 23, **PARAM_CURR_SAT** = 24,
  PARAM_CURR_DEAD_ZONE = 25, **PARAM_CUFF_ACTIVATION_FLAG** = 26, **PARAM_POWER_TE↩ NSION** = 27 }
- enum **qbmove_resolution** {
  RESOLUTION_360 = 0, **RESOLUTION_720** = 1, **RESOLUTION_1440** = 2, **RESOLUTION_2880** = 3,
  RESOLUTION_5760 = 4, **RESOLUTION_11520** = 5, **RESOLUTION_23040** = 6, **RESOLUTION_46080** = 7,
  RESOLUTION_92160 = 8 }
- enum **qbmove_input_mode** { **INPUT_MODE_EXTERNAL** = 0, **INPUT_MODE_ENCODER3** = 1 }
- enum **qbmove_control_mode** { **CONTROL_ANGLE** = 0, **CONTROL_PWM** = 1, **CONTROL_CURRENT** = 2, **CURR_AND_POS_CONTROL** = 3 }
- enum **acknowledgment_values** { **ACK_ERROR** = 0, **ACK_OK** = 1 }
- enum **data_types** {
  TYPE_FLAG = 0, **TYPE_INT8** = 1, **TYPE_UINT8** = 2, **TYPE_INT16** = 3,
  TYPE_UINT16 = 4, **TYPE_INT32** = 5, **TYPE_UINT32** = 6, **TYPE_FLOAT** = 7,
  TYPE_DOUBLE = 8 }

5.3.1 Detailed Description

Definitions for commands, parameters and packages.

Date

October 01, 2017

Author

Centro "E.Piaggio"

Copyright

This file is included in the CUFF and Stretch Pro firmware, in its libraries and applications. It contains all definitions that are necessary for the contruction of communication packages.

It includes definitions for all of the device commands, parameters and also the size of answer packages.

5.3.2 Enumeration Type Documentation

5.3.2.1 qbmove_command

`enum` **qbmove_command**

Enumerator

CMD_PING	Asks for a ping message.
CMD_SET_ZEROS	Command for setting the encoders zero position.
CMD_STORE_PARAMS	Stores all parameters in memory and loads them
CMD_STORE_DEFAULT_PARAMS	Store current parameters as factory parameters.
CMD_RESTORE_PARAMS	Restore default factory parameters.
CMD_GET_INFO	Asks for a string of information about.
CMD_SET_VALUE	Not Used.
CMD_GET_VALUE	Not Used.
CMD_BOOTLOADER	Sets the bootloader modality to update the firmware
CMD_INIT_MEM	Initialize the memory with the defalut values.
CMD_CALIBRATE	Starts the stiffness calibration of the qbMove or the hand closure and opening calibration
CMD_GET_PARAM_LIST	Command to get the parameters list or to set a defined value chosen by the use
CMD_HAND_CALIBRATE	Starts a series of opening and closures of the hand.
CMD_ACTIVATE	Command for activating/deactivating the device
CMD_GET_ACTIVATE	Command for getting device activation state
CMD_SET_INPUTS	Command for setting reference inputs.
CMD_GET_INPUTS	Command for getting reference inputs.
CMD_GET_MEASUREMENTS	Command for asking device's position measurements
CMD_GET_CURRENTS	Command for asking device's current measurements

Enumerator

CMD_GET_CURR_AND_MEAS	Command for asking device's measurements and currents
CMD_SET_POS_STIFF	Command for setting shaft position and stiffness
CMD_GET_VELOCITIES	Command for asking device's current velocity of motors and pulley
CMD_GET_COUNTERS	Command for asking device's counters (mostly used for debugging sent commands)
CMD_GET_ACCEL	Command for asking device's acceleration measurements
CMD_GET_CURR_DIFF	Command for asking device's current difference between a measured one and an estimated one (Only for SoftHand)
CMD_SET_CURR_DIFF	Command used to set current difference modality (Only for Cuff device)
CMD_SET_CUFF_INPUTS	Command used to set Cuff device inputs (Only for Cuff device)
CMD_SET_WATCHDOG	Command for setting watchdog timer or disable it
CMD_SET_BAUDRATE	Command for setting baudrate of communication

5.3.2.2 qbmove_control_mode

`enum` **`qbmove_control_mode`**

Enumerator

CONTROL_ANGLE	Classic position control.
CONTROL_PWM	Direct PWM value.
CONTROL_CURRENT	Current control (beta)
CURR_AND_POS_CONTROL	Position and current control.

5.3.2.3 qbmove_input_mode

`enum` **`qbmove_input_mode`**

Enumerator

INPUT_MODE_EXTERNAL	References through external commands (default)
INPUT_MODE_ENCODER3	Encoder 3 drives all inputs.

5.3.2.4 qbmove_parameter

`enum` **`qbmove_parameter`**

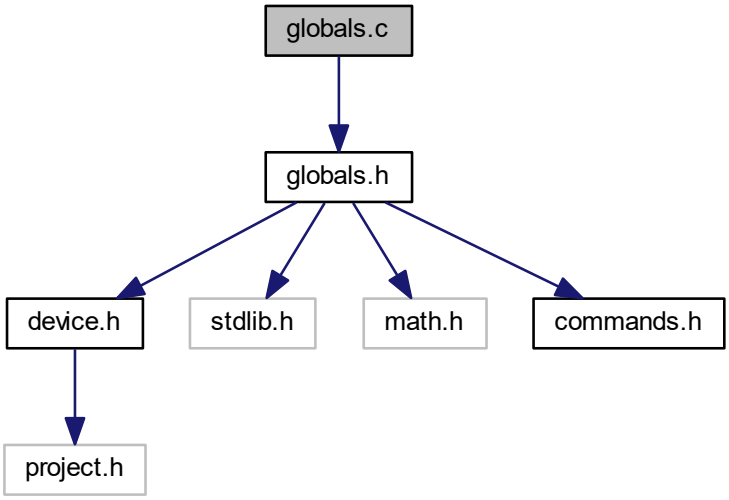
Enumerator

PARAM_ID	Device's ID number.
PARAM_PID_CONTROL	PID Control proportional constant.
PARAM_STARTUP_ACTIVATION	Start up activation byte.
PARAM_INPUT_MODE	Input mode.
PARAM_CONTROL_MODE	Choose the kind of control between position control, current control or direct PWM value input
PARAM_MEASUREMENT_OFFSET	Adds a constant offset to the measurements
PARAM_MEASUREMENT_MULTIPLIER	Adds a multiplier to the measurements
PARAM_POS_LIMIT_FLAG	Enable/disable position limiting.
PARAM_POS_LIMIT	Position limit values | int32 | int32 | int32 | int32 | | INF_LIM_1 | SUP_LIM_1 | INF_LIM_2 | SUP_LIM_2 |
PARAM_MAX_STEP_POS	Used to slow down movements for positive values.
PARAM_MAX_STEP_NEG	Used to slow down movements for negative values.
PARAM_POS_RESOLUTION	Angle resolution for inputs and measurements. Used during communication.
PARAM_CURRENT_LIMIT	Limit for absorbed current.
PARAM_PID_CURR_CONTROL	Current PID controller values.
PARAM_CURR_PROP_GAIN	Proportional gain on current difference (Only for Cuff device)
PARAM_CURR_SAT	Current difference saturation value (Only for Cuff device)
PARAM_CURR_DEAD_ZONE	Current dead zone value (Only for Cuff device)
PARAM_CUFF_ACTIVATION_FLAG	Cuff startup activation flag.
PARAM_POWER_TENSION	Device power tension.

5.4 globals.c File Reference

Global variables.

`#include <globals.h>`
Include dependency graph for globals.c:



Variables

- struct **st_ref** g_ref g_refNew **g_refOld**
- struct **st_meas** g_meas **g_measOld**
- struct **st_data g_rx**
- struct **st_mem** g_mem **c_mem**
- uint32 **timer_value**
- uint32 **timer_value0**
- int32 **dev_tension**
- uint8 **dev_pwm_limit**
- uint8 **calibration_flag**
- CYBIT **reset_last_value_flag**
- CYBIT **tension_valid**
- CYBIT **cuff_flag**
- CYBIT **interrupt_flag**
- CYBIT **watchdog_flag**
- int16 **ADC_buf** [3]
- int8 **pwm_sign** [2]

5.4.1 Detailed Description

Global variables.

Date

October 01, 2017

Author

Centro "E.Piaggio"

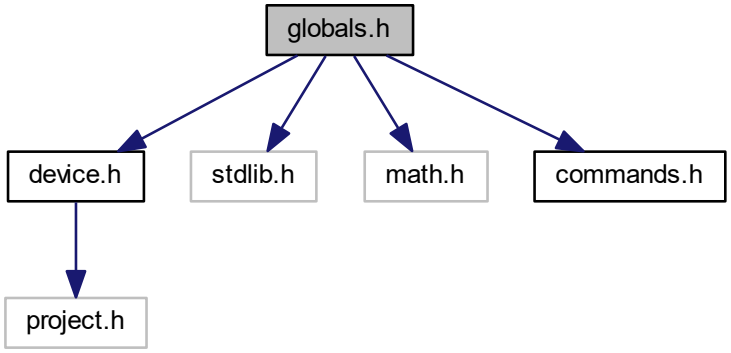
Copyright

5.5 globals.h File Reference

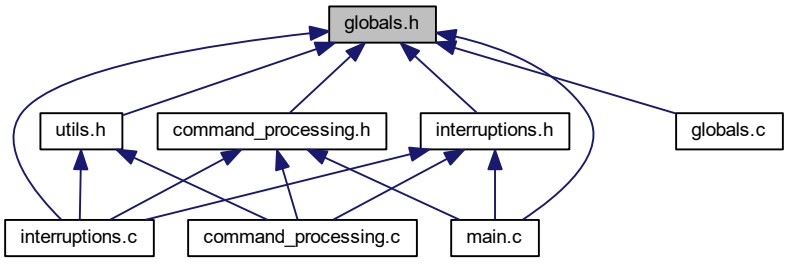
Global definitions and macros are set in this file.

```
#include <device.h>
#include "stdlib.h"
#include "math.h"
#include "commands.h"
```

Include dependency graph for globals.h:



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



Data Structures

- struct **st_ref**
- struct **st_meas**
- struct **st_data**
- struct **st_mem**

Macros

- #define **VERSION** "CUFF MOD v6.1.0"
- #define **NUM_OF_MOTORS** 2
- #define **NUM_OF_SENSORS** 3
- #define **NUM_OF_ANALOG_INPUTS** 3
- #define **NUM_OF_PARAMS** 18
- #define **PWM_MAX_VALUE** 100
- #define **PWM_DEAD** 0
- #define **POS_INTEGRAL_SAT_LIMIT** 100000
- #define **CURR_INTEGRAL_SAT_LIMIT** 100000
- #define **CALIB_CURRENT** 1000
- #define **DEFAULT_CURRENT_LIMIT** 1500
- #define **CALIBRATION_DIV** 100
- #define **DIV_INIT_VALUE** 1
- #define **DMA_BYTES_PER_BURST** 2
- #define **DMA_REQUEST_PER_BURST** 1
- #define **DMA_SRC_BASE** (CYDEV_PERIPH_BASE)
- #define **DMA_DST_BASE** (CYDEV_SRAM_BASE)
- #define **WAIT_START** 0
- #define **WAIT_ID** 1
- #define **WAIT_LENGTH** 2
- #define **RECEIVE** 3
- #define **UNLOAD** 4
- #define **FALSE** 0
- #define **TRUE** 1
- #define **DEFAULT_EEPROM_DISPLACEMENT** 8
- #define **MAX_WATCHDOG_TIMER** 250

Enumerations

- enum **calibration_status** {
  STOP = 0, **START** = 1, **CONTINUE_1** = 2, **CONTINUE_2** = 3,
  PAUSE_1 = 4, **PAUSE_2** = 5 }

Variables

- struct **st_ref** g_ref g_refNew **g_refOld**
- struct **st_meas** g_meas **g_measOld**
- struct **st_data g_rx**
- struct **st_mem** g_mem **c_mem**
- uint32 **timer_value**
- uint32 **timer_value0**
- int32 **dev_tension**
- uint8 **dev_pwm_limit**
- uint8 **calibration_flag**
- CYBIT **reset_last_value_flag**
- CYBIT **tension_valid**
- CYBIT **cuff_flag**
- CYBIT **interrupt_flag**
- CYBIT **watchdog_flag**
- int16 **ADC_buf** [3]
- int8 **pwm_sign** [2]

5.5.1 Detailed Description

Global definitions and macros are set in this file.

Date

October 01, 2017

Author

Centro "E.Piaggio"

Copyright

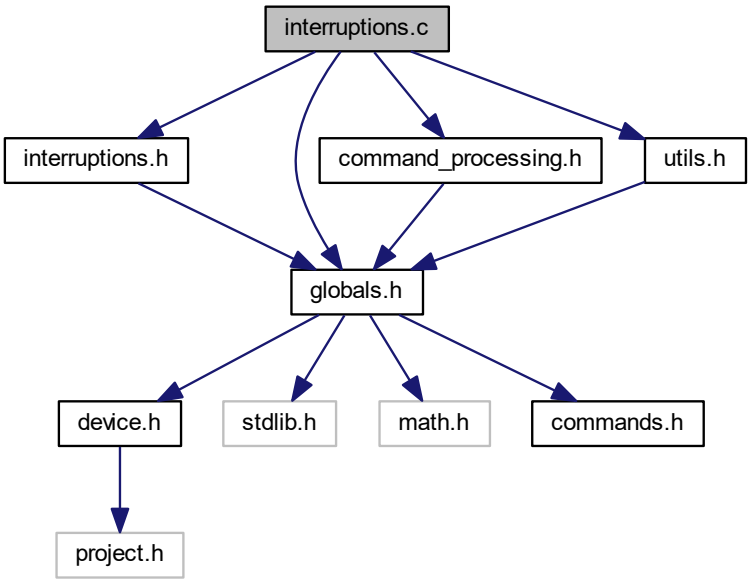
5.6 interruptions.c File Reference

Interruption functions are in this file.

```
#include <interruptions.h>
#include <command_processing.h>
#include "globals.h"
```

```
#include "utils.h"
```

Include dependency graph for interruptions.c:



Functions

- **CY_ISR** (ISR_WATCHDOG_Handler)
- **CY_ISR** (ISR_RS485_RX_ExInterrupt)
- void **interrupt_manager** ()
- void **function_scheduler** (void)
- void **motor_control** (const uint8 idx)
- void **analog_read_end** ()
- void **encoder_reading** (const uint8 idx, const uint8 flag)
- void **calibration** ()
- void **pwm_limit_search** ()

Variables

- CYCODE uint8 **pwm_preload_values** [29]
- CYCODE uint8 **hitech_pwm_preload_values** [36]
- CYCODE uint8 **hitech_pwm_preload_values_6v** [32]

5.6.1 Detailed Description

Interruption functions are in this file.

Date

October 01, 2017

Author

Centro "E.Piaggio"

Copyright

5.6.2 Variable Documentation

5.6.2.1 pwm_preload_values

```
CYCODE uint8 pwm_preload_values[29]
```

Initial value:

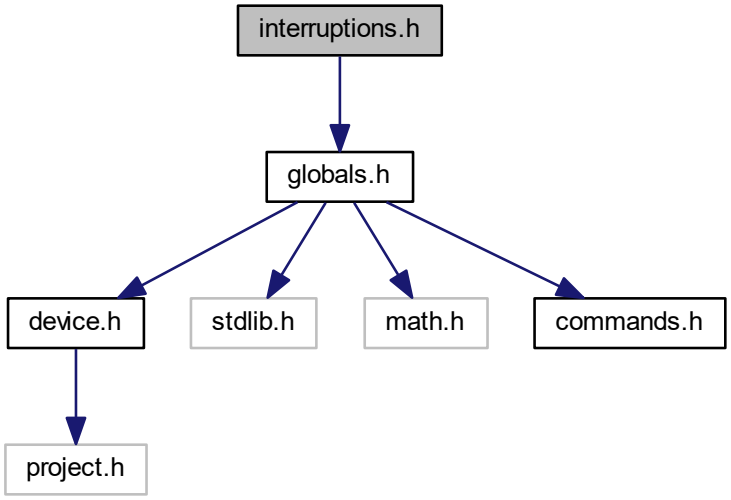
```
= {100,
                                        100,
                                        100,
                                         76,
                                         74,
                                         72,
                                         70,
                                         68,
                                         67,
                                         65,
                                         64,
                                         63,
                                         62,
                                         61,
                                         60,
                                         59,
                                         58,
                                         57,
                                         56,
                                         56,
                                         55,
                                         54,
                                         54,
                                         53,
                                         52,
                                         52,
                                         52,
                                         51,
                                         51}
```

5.7 interruptions.h File Reference

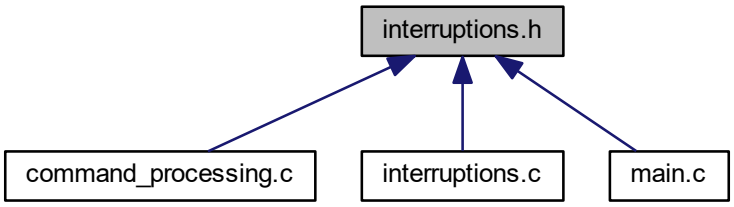
Interruptions header file.

```
#include <globals.h>
```

Include dependency graph for interruptions.h:



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



Functions

- **CY_ISR_PROTO** (ISR_RS485_RX_ExInterrupt)
- **CY_ISR_PROTO** (ISR_WATCHDOG_Handler)
- void **function_scheduler** (void)
- void **encoder_reading** (const uint8, const uint8)
- void **motor_control** (const uint8)
- void **analog_read_end** ()
- void **calibration** (void)
- void **pwm_limit_search** ()
- void **interrupt_manager** ()

5.7.1 Detailed Description

Interruptions header file.

Date

October 01, 2017

Author

Centro "E.Piaggio"

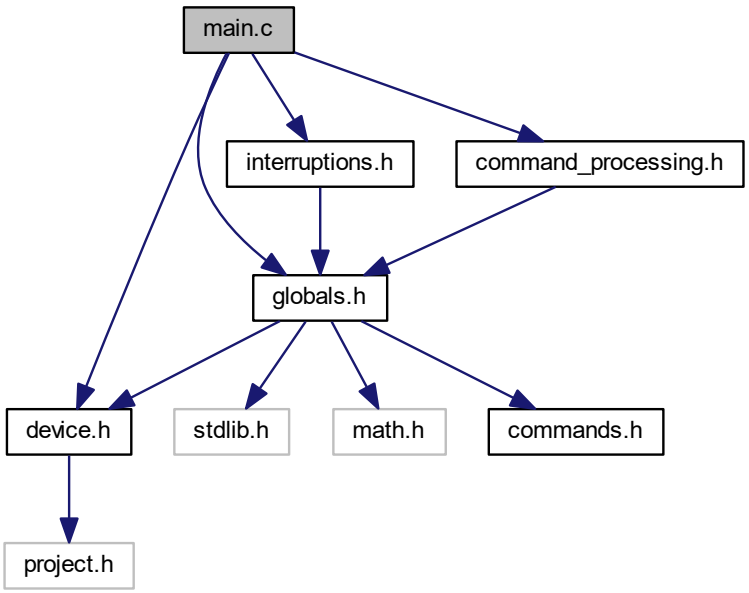
Copyright

5.8 main.c File Reference

Firmware main file.

```
#include <device.h>
#include <globals.h>
#include <interruptions.h>
#include <command_processing.h>
```

Include dependency graph for main.c:



Functions

- int **main** ()

5.8.1 Detailed Description

Firmware main file.

Date

October 01, 2017

Author

Centro "E.Piaggio"

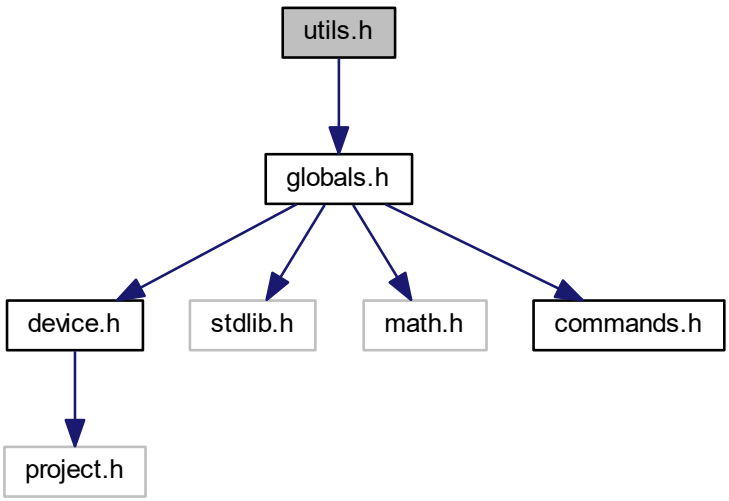
Copyright

5.9 utils.h File Reference

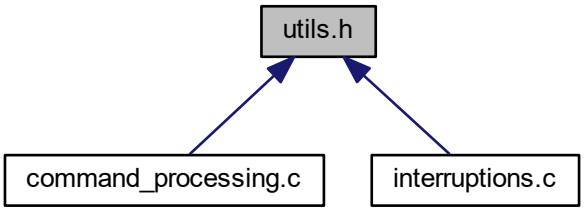
Definition of utility functions.

```
#include <globals.h>
```

Include dependency graph for utils.h:



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



Macros

- #define **SIGN**(A) (((A) > 0) ? (1) : ((((A) < 0) ? (-1) : (0))))

Functions

- int32 **filter_i1** (int32 value)
- int32 **filter_i2** (int32 value)
- int32 **filter_vel_1** (int32 value)
- int32 **filter_vel_2** (int32 value)
- int32 **filter_vel_3** (int32 value)
- uint8 **LCRChecksum** (uint8 *data_array, uint8 data_length)
- CYBIT **check_enc_data** (const uint32 *)

5.9.1 Detailed Description

Definition of utility functions.

Declaration of utility functions.

Date

October 01, 2017

Author

Centro "E.Piaggio"

Copyright

Date

Dic. 1, 2015

Author

qbrobotics

Copyright

Index

cmd_get_measurements
    command_processing.c, 10
    command_processing.h, 13
command_processing.c, 9
    cmd_get_measurements, 10
    memInit, 11
    memRecall, 11
    memRestore, 11
    memStore, 11
command_processing.h, 12
    cmd_get_measurements, 13
    memInit, 13
    memRecall, 14
    memRestore, 14
    memStore, 14
commands.h, 14
    qbmove_command, 16
    qbmove_control_mode, 17
    qbmove_input_mode, 17
    qbmove_parameter, 17

globals.c, 18
globals.h, 20

interruptions.c, 22
    pwm_preload_values, 24
interruptions.h, 25

main.c, 26
memInit
    command_processing.c, 11
    command_processing.h, 13
memRecall
    command_processing.c, 11
    command_processing.h, 14
memRestore
    command_processing.c, 11
    command_processing.h, 14
memStore
    command_processing.c, 11
    command_processing.h, 14

pwm_preload_values
    interruptions.c, 24

qbmove_command
    commands.h, 16
qbmove_control_mode
    commands.h, 17
qbmove_input_mode
    commands.h, 17
qbmove_parameter
    commands.h, 17

st_data, 7
st_meas, 7
st_mem, 8
st_ref, 8

utils.h, 27