Micras Firmware

Generated by Doxygen 1.9.1

Chapter 1

Micras

Chapter 2

Todo List

Member proxy::Imu::get_orientation (Axis axis) const

implement function using sensior fusion

Chapter 3

Namespace Index

3.1 Namespace List

Here is a list of all namespaces with brief descriptions:

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 files with brief descriptions:

Chapter 6

Namespace Documentation

6.1 hal Namespace Reference

Classes

- class AdcDma

  Class to handle ADC peripheral on STM32 microcontrollers using DMA.
- class Crc

  Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.
- class Encoder

  Class to handle encoder peripheral on STM32 microcontrollers.
- class Flash

  Class to handle flash memory on STM32 microcontrollers.
- class Gpio

  Class for controlling GPIO pins on STM32 microcontrollers.
- class Mcu

  Microcontroller unit class.
- class Pwm

  Class to handle PWM peripheral on STM32 microcontrollers.
- class PwmDma

  Class to handle PWM peripheral on STM32 microcontrollers using DMA.
- class Spi

  Class to handle SPI peripheral on STM32 microcontrollers.
- class Timer

  Class to handle timer peripheral on STM32 microcontrollers.

6.2 proxy Namespace Reference

Classes

- class Argb

  Class for controlling an addressable RGB LED.
- class Battery

  Class for getting the battery voltage.

- class Button

  Class for controlling a button.
- class Buzzer

  Class for controlling a buzzer.
- class CurrentSensors

  Class for controlling CurrentSensors.
- class DipSwitch

  Class for controlling a dip switch.
- class DistanceSensors

  Class for controlling DistanceSensors.
- class Fan

  Class for controlling the fan driver.
- class Imu

  Class to handle IMU peripheral on STM32 microcontrollers.
- class Led

  Class for controlling an LED.
- class Locomotion

  Class for controlling the locomotion driver.
- class RotarySensor

  Class to handle rotary sensor peripheral on STM32 microcontrollers.
- class Storage

  Class for controlling the storage.
- class TorqueSensors

  Class for controlling TorqueSensors.

Variables

- template<typename T >
  concept Fundamental = std::is_fundamental<T>::value

6.2.1 Variable Documentation

6.2.1.1 Fundamental

```
template<typename T >
concept proxy::Fundamental = std::is_fundamental<T>::value
```

Chapter 7

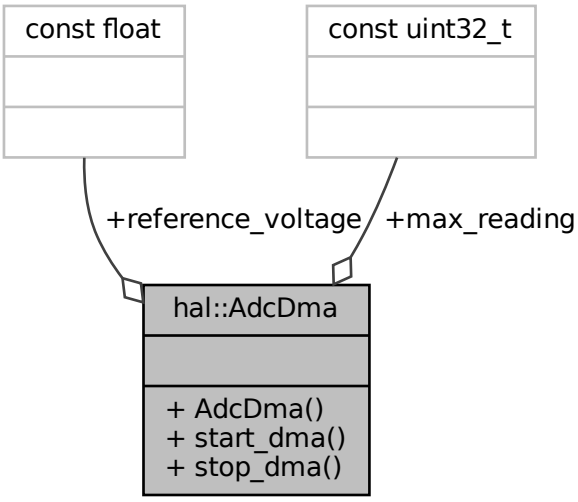
Class Documentation

7.1 hal::AdcDma Class Reference

Class to handle ADC peripheral on STM32 microcontrollers using DMA.

`#include <adc_dma.hpp>`

Collaboration diagram for hal::AdcDma:



Classes

- struct Config

  Configuration structure for ADC DMA.

Public Member Functions

- AdcDma (const Config &config)

  Construct a new AdcDma object.
- void start_dma (uint32_t buffer[ ], uint32_t size)

  Enable ADC, start conversion of regular group and transfer result through DMA.
- void stop_dma ()

  Stop ADC conversion of regular group (and injected group in case of auto_injection mode)

Public Attributes

- const uint32_t max_reading

  Maximum ADC reading.
- const float reference_voltage

  Reference voltage for the ADC measurement.

7.1.1 Detailed Description

Class to handle ADC peripheral on STM32 microcontrollers using DMA.

7.1.2 Constructor & Destructor Documentation

7.1.2.1 AdcDma()

```
hal::AdcDma::AdcDma (
            const Config & config )  [explicit]
```

Construct a new AdcDma object.

Parameters

config	ADC DMA configuration struct

7.1.3 Member Function Documentation

7.1.3.1 start_dma()

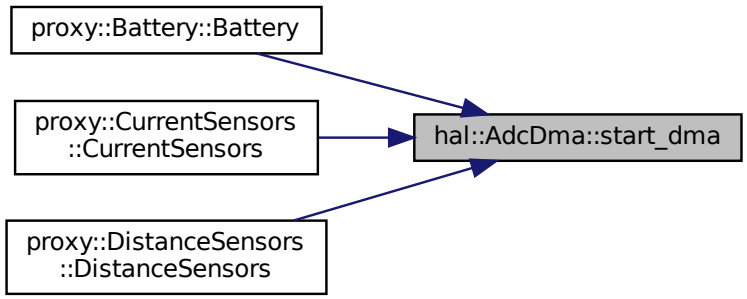
```
void hal::AdcDma::start_dma (
            uint32_t buffer[],
            uint32_t size )
```

Enable ADC, start conversion of regular group and transfer result through DMA.

Parameters

buffer	Destination Buffer address
size	Number of data to be transferred from ADC DMA peripheral to memory

Here is the caller graph for this function:



7.1.3.2 stop_dma()

```
void hal::AdcDma::stop_dma ( )
```

Stop ADC conversion of regular group (and injected group in case of auto_injection mode)

7.1.4 Member Data Documentation

7.1.4.1 max_reading

```
const uint32_t hal::AdcDma::max_reading
```

Maximum ADC reading.

7.1.4.2 reference_voltage

```
const float hal::AdcDma::reference_voltage
```

Reference voltage for the ADC measurement.

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

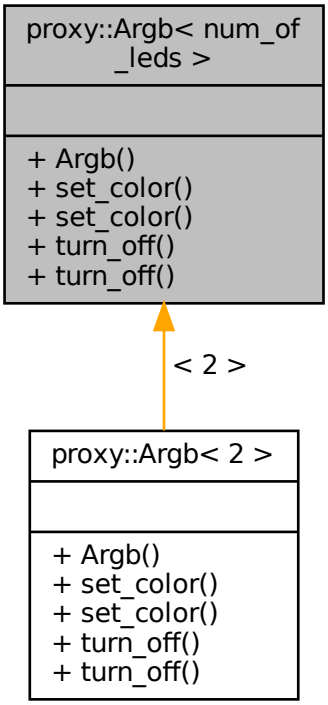
- inc/hal/adc_dma.hpp
- src/hal/adc_dma.cpp

7.2 proxy::Argb< num_of_leds > Class Template Reference

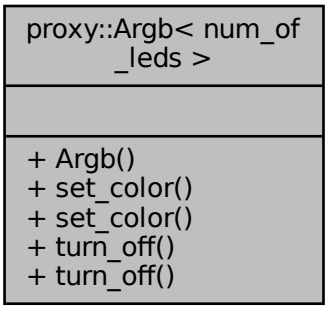
Class for controlling an addressable RGB LED.

`#include <argb.hpp>`

Inheritance diagram for proxy::Argb< num_of_leds >:



Collaboration diagram for proxy::Argb< num_of_leds >:



Classes

- struct Color

  Structure for storing color information.
- struct Config

  Configuration structure for the addressable RGB LED.

Public Member Functions

- Argb (const Config &config)

  Constructor for the Argb class.
- void set_color (const Color &color, uint8_t index)

  Set the color of the ARGB at the specified index.
- void set_color (const Color &color)

  Set the color of all ARGBs.
- void turn_off (uint8_t index)

  Turn off the ARGB at the specified index.
- void turn_off ()

  Turn off all ARGBs.

7.2.1 Detailed Description

template<uint8_t num_of_leds>
class proxy::Argb< num_of_leds >

Class for controlling an addressable RGB LED.

7.2.2 Constructor & Destructor Documentation

7.2.2.1 Argb()

```
template<uint8_t num_of_leds>
proxy::Argb< num_of_leds >::Argb (
            const Config & config )  [explicit]
```

Constructor for the Argb class.

Parameters

config	Configuration for the addressable RGB LED

7.2.3 Member Function Documentation

7.2.3.1 set_color() [1/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::set_color (
            const Color & color )
```

Set the color of all ARGBs.

Parameters

color	The color to set all ARGBs to

7.2.3.2 set_color() [2/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::set_color (
            const Color & color,
            uint8_t index )
```

Set the color of the ARGB at the specified index.

Parameters

index	The index of the ARGB to set the color of
color	The color to set the ARGB to

7.2.3.3 turn_off() [1/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::turn_off
```

Turn off all ARGBs.

7.2.3.4 turn_off() [2/2]

```
template<uint8_t num_of_leds>
void proxy::Argb< num_of_leds >::turn_off (
            uint8_t index )
```

Turn off the ARGB at the specified index.

Parameters

index	The index of the ARGB to turn off

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

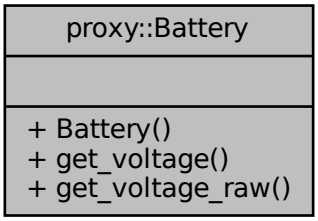
- inc/proxy/argb.hpp
- src/proxy/argb.cpp

7.3 proxy::Battery Class Reference

Class for getting the battery voltage.

`#include <battery.hpp>`

Collaboration diagram for proxy::Battery:



Classes

- struct Config

  Configuration structure for the battery.

Public Member Functions

- Battery (const Config &config)

  Constructor for the Battery class.
- float get_voltage () const

  Get the battery voltage.
- uint32_t get_voltage_raw () const

  Get the raw reading from the battery.

7.3.1 Detailed Description

Class for getting the battery voltage.

7.3.2 Constructor & Destructor Documentation

7.3.2.1 Battery()

```
proxy::Battery::Battery (
            const Config & config )  [explicit]
```

Constructor for the Battery class.

Parameters

config	Configuration for the battery

Here is the call graph for this function:

7.3.3 Member Function Documentation

7.3.3.1 get_voltage()

```
float proxy::Battery::get_voltage ( ) const
```

Get the battery voltage.

Returns

float Battery voltage in volts

7.3.3.2 get_voltage_raw()

```
uint32_t proxy::Battery::get_voltage_raw ( ) const
```

Get the raw reading from the battery.

Returns

uint32_t Raw reading from the battery

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

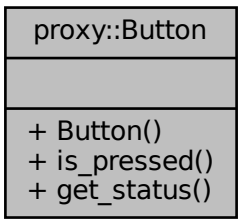
- inc/proxy/battery.hpp
- src/proxy/battery.cpp

7.4 proxy::Button Class Reference

Class for controlling a button.

`#include <button.hpp>`

Collaboration diagram for proxy::Button:



Classes

- struct Config

  Configuration structure for button.

Public Types

- enum Status { NO_PRESS , SHORT_PRESS , LONG_PRESS , EXTRA_LONG_PRESS }

  Enum for button status.
- enum PullResistor { PULL_UP , PULL_DOWN }

  Enum for button pull resistor.

Public Member Functions

- Button (const Config &config)

  Constructor for Button class.
- bool is_pressed ()

  Check if button is pressed.
- Status get_status ()

  Get button status.

7.4.1 Detailed Description

Class for controlling a button.

7.4.2 Member Enumeration Documentation

7.4.2.1 PullResistor

```
enum proxy::Button::PullResistor
```

Enum for button pull resistor.

Enumerator

PULL_UP	
PULL_DOWN	

7.4.2.2 Status

```
enum proxy::Button::Status
```

Enum for button status.

Enumerator

NO_PRESS	
SHORT_PRESS	
LONG_PRESS	
EXTRA_LONG_PRESS	

7.4.3 Constructor & Destructor Documentation

7.4.3.1 Button()

```
proxy::Button::Button (
            const Config & config )  [explicit]
```

Constructor for Button class.

Parameters

config	Button configuration

7.4.4 Member Function Documentation

7.4.4.1 get_status()

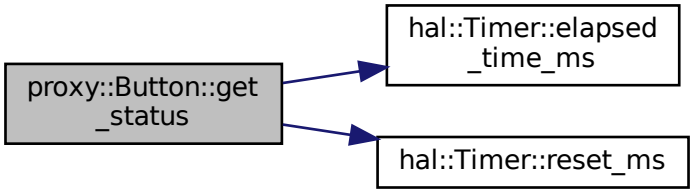
```
Button::Status proxy::Button::get_status ( )
```

Get button status.

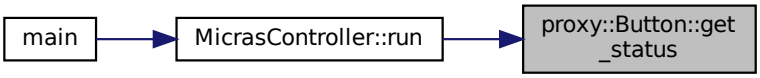
Returns

Status Button status

Here is the call graph for this function:



Here is the caller graph for this function:



7.4.4.2 is_pressed()

```
bool proxy::Button::is_pressed ( )
```

Check if button is pressed.

Returns

bool True if button is pressed, false otherwise

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

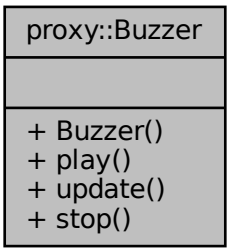
- inc/proxy/button.hpp
- src/proxy/button.cpp

7.5 proxy::Buzzer Class Reference

Class for controlling a buzzer.

`#include <buzzer.hpp>`

Collaboration diagram for proxy::Buzzer:



Classes

- struct Config

  Configuration structure for the buzzer.

Public Member Functions

- Buzzer (const Config &config)

  Constructor for the Buzzer class.
- void play (uint32_t frequency, uint32_t duration=0)

  Play a tone for a duration.
- void update ()

  Update the buzzer state.
- void stop ()

  Stop the buzzer sound.

7.5.1 Detailed Description

Class for controlling a buzzer.

7.5.2 Constructor & Destructor Documentation

7.5.2.1 Buzzer()

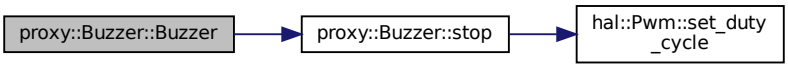
```
proxy::Buzzer::Buzzer (
            const Config & config )  [explicit]
```

Constructor for the Buzzer class.

Parameters

config	Configuration for the buzzer

Here is the call graph for this function:



7.5.3 Member Function Documentation

7.5.3.1 play()

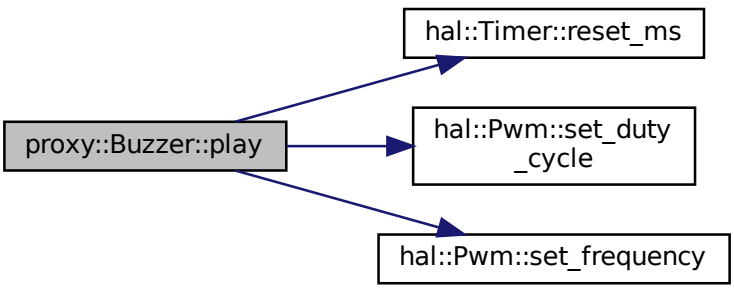
```
void proxy::Buzzer::play (
            uint32_t frequency,
            uint32_t duration = 0 )
```

Play a tone for a duration.

Parameters

frequency	Buzzer sound frequency in Hz
duration	Duration of the sound in ms

Here is the call graph for this function:

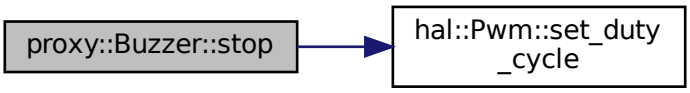


7.5.3.2 stop()

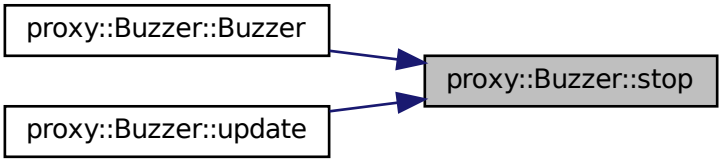
```
void proxy::Buzzer::stop ( )
```

Stop the buzzer sound.

Here is the call graph for this function:



Here is the caller graph for this function:

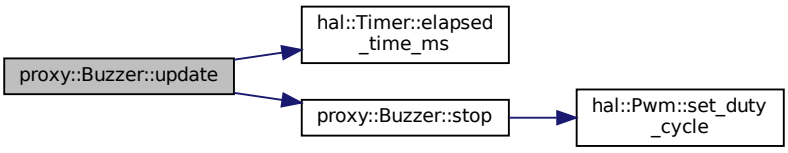


7.5.3.3 update()

```
void proxy::Buzzer::update ( )
```

Update the buzzer state.

Here is the call graph for this function:



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

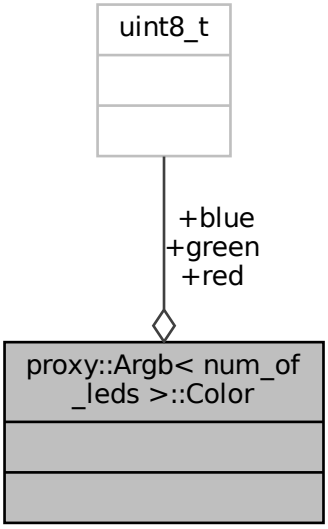
- inc/proxy/buzzer.hpp
- src/proxy/buzzer.cpp

7.6 proxy::Argb< num_of_leds >::Color Struct Reference

Structure for storing color information.

`#include <argb.hpp>`

Collaboration diagram for proxy::Argb< num_of_leds >::Color:



Public Attributes

- uint8_t red
- uint8_t green
- uint8_t blue

7.6.1 Detailed Description

template<uint8_t num_of_leds>
struct proxy::Argb< num_of_leds >::Color

Structure for storing color information.

7.6.2 Member Data Documentation

7.6.2.1 blue

```
template<uint8_t num_of_leds>
uint8_t proxy::Argb< num_of_leds >::Color::blue
```

7.6.2.2 green

```
template<uint8_t num_of_leds>
uint8_t proxy::Argb< num_of_leds >::Color::green
```

7.6.2.3 red

```
template<uint8_t num_of_leds>
uint8_t proxy::Argb< num_of_leds >::Color::red
```

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

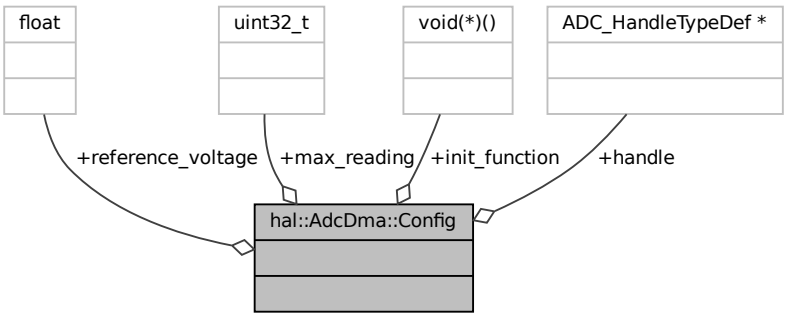
- inc/proxy/argb.hpp

7.7 hal::AdcDma::Config Struct Reference

Configuration structure for ADC DMA.

`#include <adc_dma.hpp>`

Collaboration diagram for hal::AdcDma::Config:



Public Attributes

- ADC_HandleTypeDef * handle
- void(* init_function)()
- uint32_t max_reading
- float reference_voltage

7.7.1 Detailed Description

Configuration structure for ADC DMA.

7.7.2 Member Data Documentation

7.7.2.1 handle

```
ADC_HandleTypeDef* hal::AdcDma::Config::handle
```

7.7.2.2 init_function

```
void(* hal::AdcDma::Config::init_function) ()
```

7.7.2.3 max_reading

```
uint32_t hal::AdcDma::Config::max_reading
```

7.7.2.4 reference_voltage

```
float hal::AdcDma::Config::reference_voltage
```

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

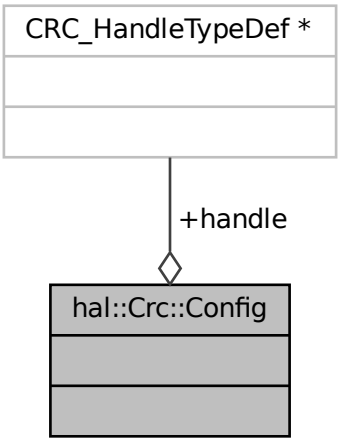
- inc/hal/adc_dma.hpp

7.8 hal::Crc::Config Struct Reference

CRC configuration struct.

`#include <crc.hpp>`

Collaboration diagram for hal::Crc::Config:



Public Attributes

- CRC_HandleTypeDef * handle

7.8.1 Detailed Description

CRC configuration struct.

7.8.2 Member Data Documentation

7.8.2.1 handle

`CRC_HandleTypeDef* hal::Crc::Config::handle`

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

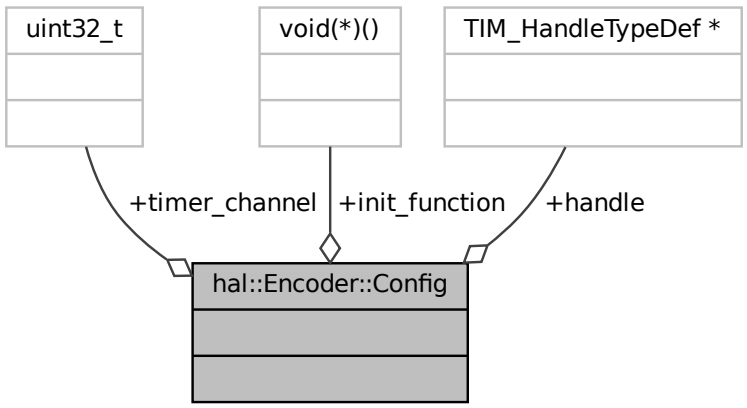
- inc/hal/crc.hpp

7.9 hal::Encoder::Config Struct Reference

Encoder configuration struct.

`#include <encoder.hpp>`

Collaboration diagram for hal::Encoder::Config:



Public Attributes

- TIM_HandleTypeDef * handle
- void(* init_function)()
- uint32_t timer_channel

7.9.1 Detailed Description

Encoder configuration struct.

7.9.2 Member Data Documentation

7.9.2.1 handle

`TIM_HandleTypeDef* hal::Encoder::Config::handle`

7.9.2.2 init_function

```
void(* hal::Encoder::Config::init_function) ()
```

7.9.2.3 timer_channel

```
uint32_t hal::Encoder::Config::timer_channel
```

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

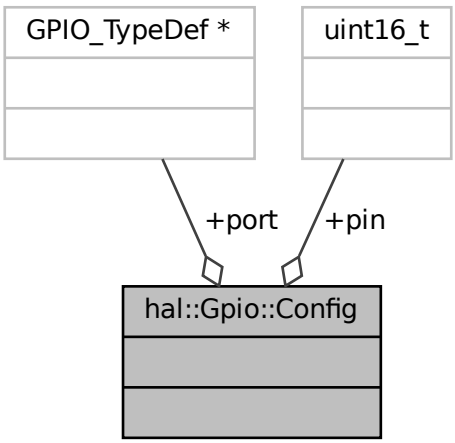
- inc/hal/encoder.hpp

7.10 hal::Gpio::Config Struct Reference

Configuration structure for GPIO pin.

```
#include <gpio.hpp>
```

Collaboration diagram for hal::Gpio::Config:



Public Attributes

- GPIO_TypeDef * port
- uint16_t pin

7.10.1 Detailed Description

Configuration structure for GPIO pin.

7.10.2 Member Data Documentation

7.10.2.1 pin

```
uint16_t hal::Gpio::Config::pin
```

7.10.2.2 port

```
GPIO_TypeDef* hal::Gpio::Config::port
```

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

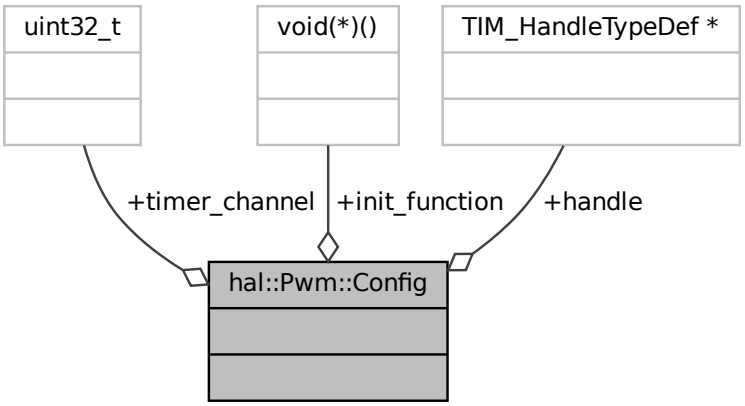
- inc/hal/gpio.hpp

7.11 hal::Pwm::Config Struct Reference

PWM configuration struct.

```
#include <pwm.hpp>
```

Collaboration diagram for hal::Pwm::Config:



Public Attributes

- TIM_HandleTypeDef * handle
- void(* init_function)()
- uint32_t timer_channel

7.11.1 Detailed Description

PWM configuration struct.

7.11.2 Member Data Documentation

7.11.2.1 handle

```
TIM_HandleTypeDef* hal::Pwm::Config::handle
```

7.11.2.2 init_function

```
void(* hal::Pwm::Config::init_function) ()
```

7.11.2.3 timer_channel

```
uint32_t hal::Pwm::Config::timer_channel
```

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

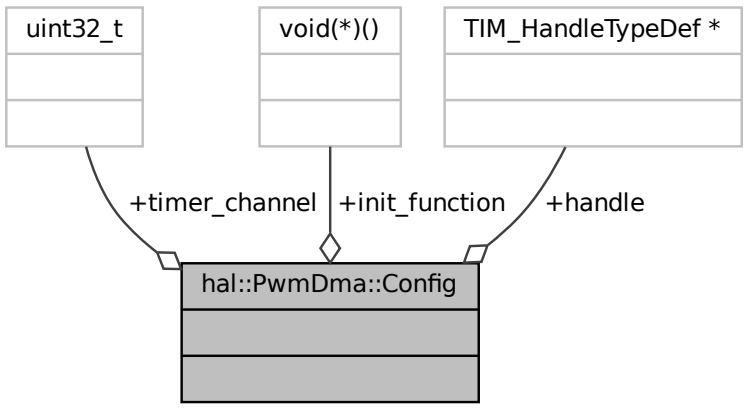
- inc/hal/pwm.hpp

7.12 hal::PwmDma::Config Struct Reference

PWM configuration struct.

`#include <pwm_dma.hpp>`

Collaboration diagram for hal::PwmDma::Config:



Public Attributes

- TIM_HandleTypeDef * handle
- void(* init_function)()
- uint32_t timer_channel

7.12.1 Detailed Description

PWM configuration struct.

7.12.2 Member Data Documentation

7.12.2.1 handle

`TIM_HandleTypeDef* hal::PwmDma::Config::handle`

7.12.2.2 init_function

```
void(* hal::PwmDma::Config::init_function) ()
```

7.12.2.3 timer_channel

```
uint32_t hal::PwmDma::Config::timer_channel
```

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

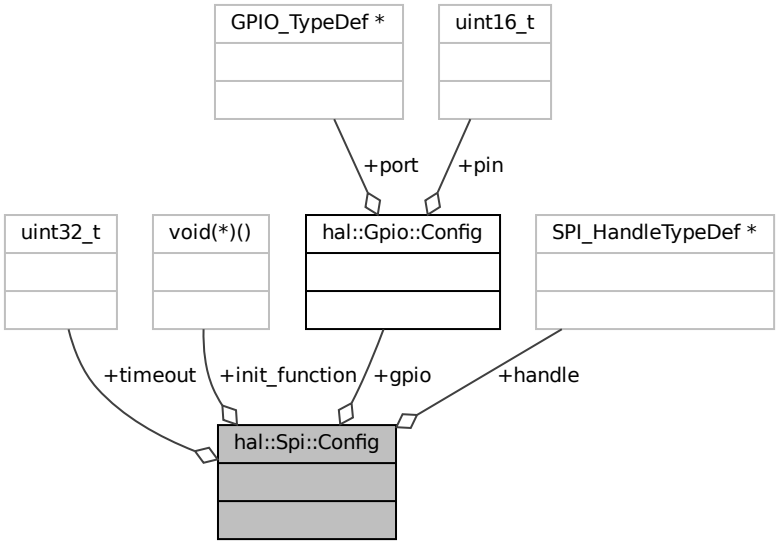
- inc/hal/pwm_dma.hpp

7.13 hal::Spi::Config Struct Reference

SPI configuration struct.

```
#include <spi.hpp>
```

Collaboration diagram for hal::Spi::Config:



Public Attributes

- SPI_HandleTypeDef * handle
- void(* init_function)()
- hal::Gpio::Config gpio
- uint32_t timeout

7.13.1 Detailed Description

SPI configuration struct.

7.13.2 Member Data Documentation

7.13.2.1 gpio

```
hal::Gpio::Config hal::Spi::Config::gpio
```

7.13.2.2 handle

```
SPI_HandleTypeDef* hal::Spi::Config::handle
```

7.13.2.3 init_function

```
void(* hal::Spi::Config::init_function) ()
```

7.13.2.4 timeout

```
uint32_t hal::Spi::Config::timeout
```

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

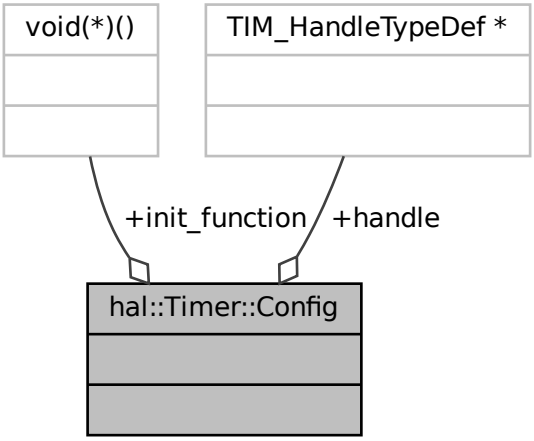
- inc/hal/spi.hpp

7.14 hal::Timer::Config Struct Reference

Timer configuration struct.

`#include <timer.hpp>`

Collaboration diagram for hal::Timer::Config:



Public Attributes

- TIM_HandleTypeDef * handle
- void(* init_function)()

7.14.1 Detailed Description

Timer configuration struct.

7.14.2 Member Data Documentation

7.14.2.1 handle

`TIM_HandleTypeDef* hal::Timer::Config::handle`

7.14.2.2 init_function

```
void(* hal::Timer::Config::init_function) ()
```

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

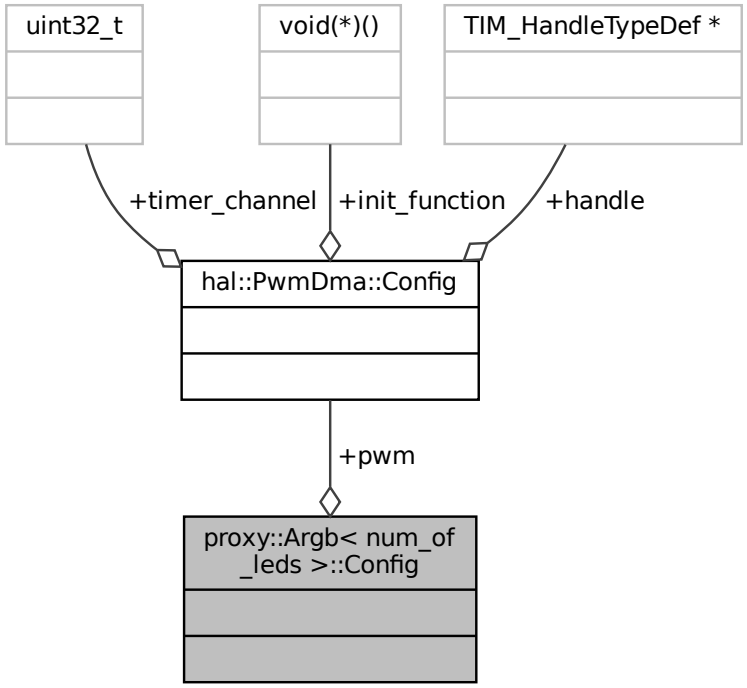
- inc/hal/timer.hpp

7.15 proxy::Argb< num_of_leds >::Config Struct Reference

Configuration structure for the addressable RGB LED.

```
#include <argb.hpp>
```

Collaboration diagram for proxy::Argb< num_of_leds >::Config:



Public Attributes

- hal::PwmDma::Config pwm

7.15.1 Detailed Description

template<uint8_t num_of_leds>
struct proxy::Argb< num_of_leds >::Config

Configuration structure for the addressable RGB LED.

7.15.2 Member Data Documentation

7.15.2.1 pwm

```
template<uint8_t num_of_leds>
hal::PwmDma::Config proxy::Argb< num_of_leds >::Config::pwm
```

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

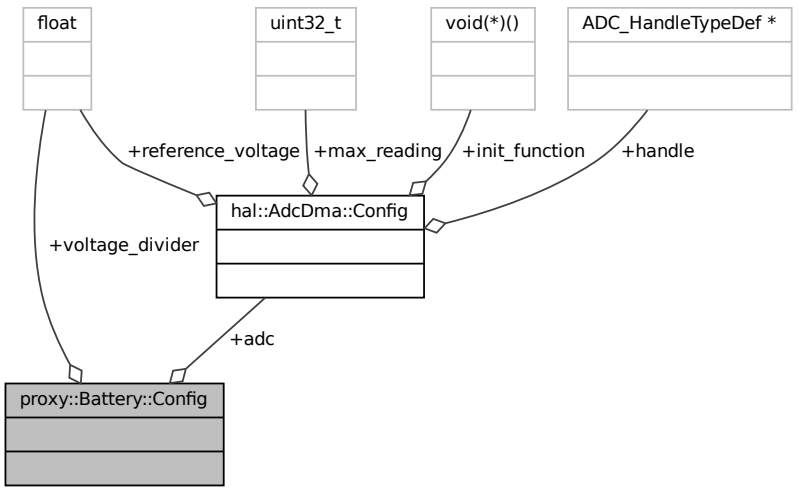
- inc/proxy/argb.hpp

7.16 proxy::Battery::Config Struct Reference

Configuration structure for the battery.

```
#include <battery.hpp>
```

Collaboration diagram for proxy::Battery::Config:



Public Attributes

- hal::AdcDma::Config adc
- float voltage_divider

7.16.1 Detailed Description

Configuration structure for the battery.

7.16.2 Member Data Documentation

7.16.2.1 adc

```
hal::AdcDma::Config proxy::Battery::Config::adc
```

7.16.2.2 voltage_divider

```
float proxy::Battery::Config::voltage_divider
```

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

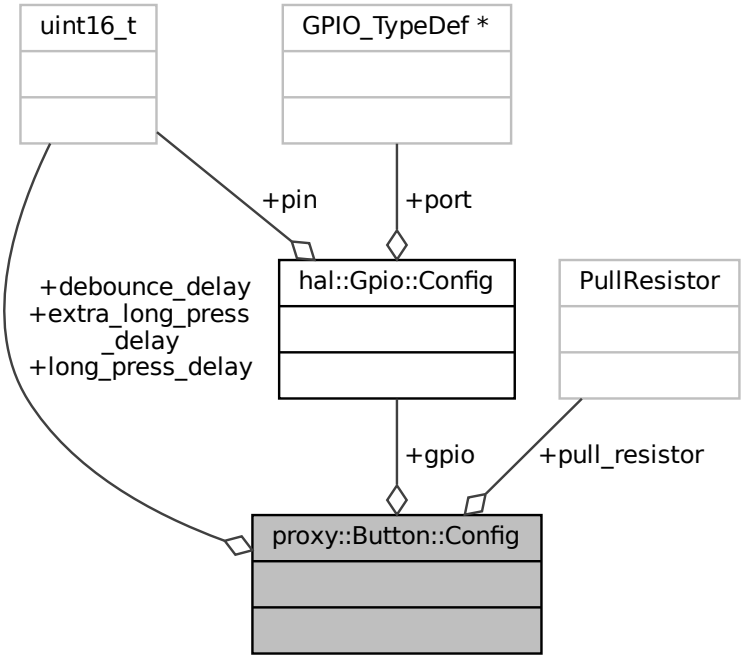
- inc/proxy/battery.hpp

7.17 proxy::Button::Config Struct Reference

Configuration structure for button.

```
#include <button.hpp>
```

Collaboration diagram for proxy::Button::Config:



Public Attributes

- hal::Gpio::Config gpio { }
- PullResistor pull_resistor { }
- uint16_t debounce_delay = 10
- uint16_t long_press_delay = 1000
- uint16_t extra_long_press_delay = 5000

7.17.1 Detailed Description

Configuration structure for button.

7.17.2 Member Data Documentation

7.17.2.1 debounce_delay

```
uint16_t proxy::Button::Config::debounce_delay = 10
```

7.17.2.2 extra_long_press_delay

```
uint16_t proxy::Button::Config::extra_long_press_delay = 5000
```

7.17.2.3 gpio

```
hal::Gpio::Config proxy::Button::Config::gpio { }
```

7.17.2.4 long_press_delay

```
uint16_t proxy::Button::Config::long_press_delay = 1000
```

7.17.2.5 pull_resistor

```
PullResistor proxy::Button::Config::pull_resistor { }
```

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

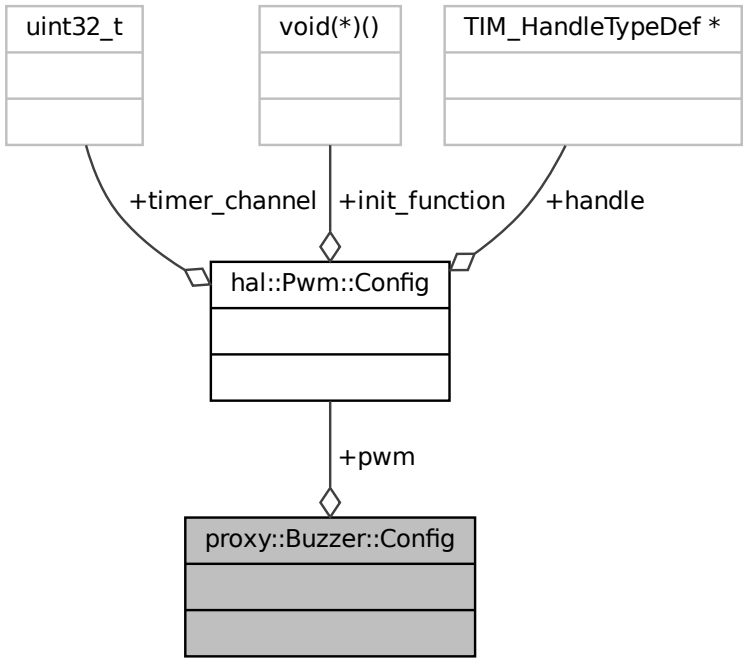
- inc/proxy/button.hpp

7.18 proxy::Buzzer::Config Struct Reference

Configuration structure for the buzzer.

```
#include <buzzer.hpp>
```

Collaboration diagram for proxy::Buzzer::Config:



Public Attributes

- hal::Pwm::Config pwm

7.18.1 Detailed Description

Configuration structure for the buzzer.

7.18.2 Member Data Documentation

7.18.2.1 pwm

`hal::Pwm::Config proxy::Buzzer::Config::pwm`

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

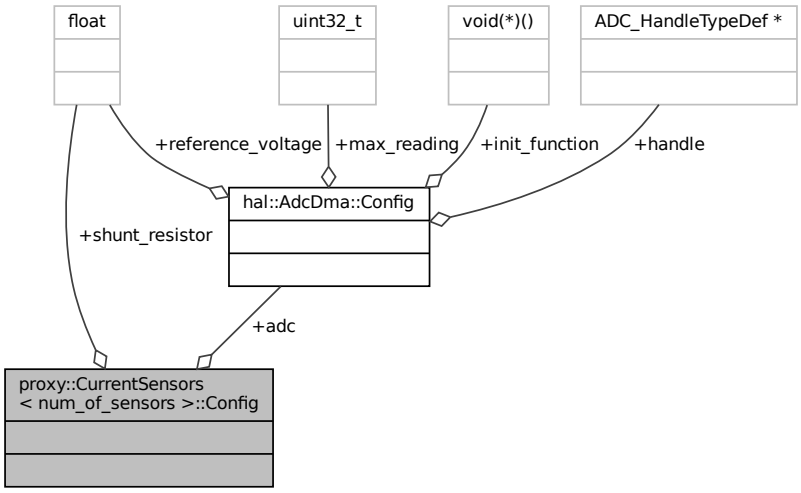
- inc/proxy/buzzer.hpp

7.19 proxy::CurrentSensors< num_of_sensors >::Config Struct Reference

Configuration structure for current sensors.

`#include <current_sensors.hpp>`

Collaboration diagram for proxy::CurrentSensors< num_of_sensors >::Config:



Public Attributes

- hal::AdcDma::Config adc
- float shunt_resistor

7.19.1 Detailed Description

template<uint8_t num_of_sensors>
struct proxy::CurrentSensors< num_of_sensors >::Config

Configuration structure for current sensors.

7.19.2 Member Data Documentation

7.19.2.1 adc

```
template<uint8_t num_of_sensors>
hal::AdcDma::Config proxy::CurrentSensors< num_of_sensors >::Config::adc
```

7.19.2.2 shunt_resistor

```
template<uint8_t num_of_sensors>
float proxy::CurrentSensors< num_of_sensors >::Config::shunt_resistor
```

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

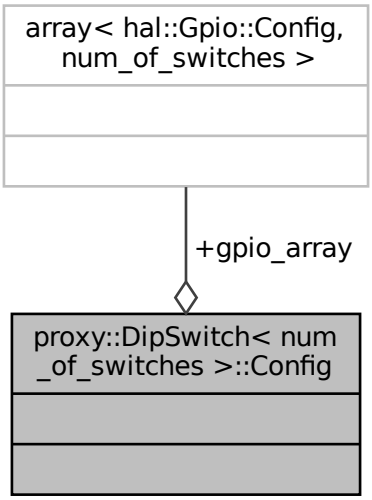
- inc/proxy/current_sensors.hpp

7.20 proxy::DipSwitch< num_of_switches >::Config Struct Reference

Configuration struct for DipSwitch.

```
#include <dip_switch.hpp>
```

Collaboration diagram for proxy::DipSwitch< num_of_switches >::Config:



Public Attributes

- std::array< hal::Gpio::Config, num_of_switches > gpio_array

7.20.1 Detailed Description

template< uint8_t num_of_switches >
struct proxy::DipSwitch< num_of_switches >::Config

Configuration struct for DipSwitch.

7.20.2 Member Data Documentation

7.20.2.1 gpio_array

```
template<uint8_t num_of_switches>
std::array<hal::Gpio::Config, num_of_switches> proxy::DipSwitch< num_of_switches >::Config↩
::gpio_array
```

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

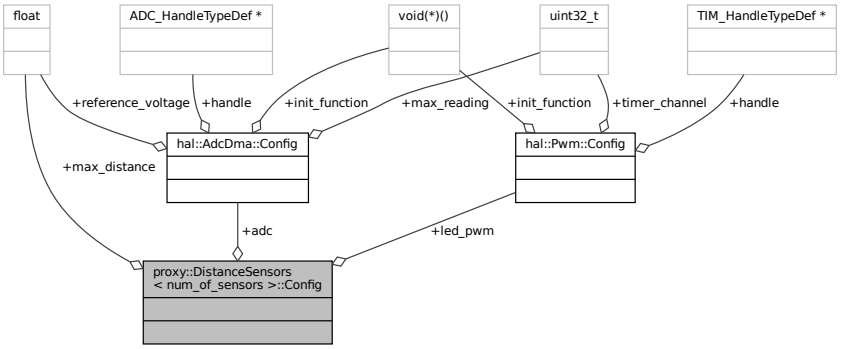
- inc/proxy/dip_switch.hpp

7.21 proxy::DistanceSensors< num_of_sensors >::Config Struct Reference

Configuration structure for distance sensors.

```
#include <distance_sensors.hpp>
```

Collaboration diagram for proxy::DistanceSensors< num_of_sensors >::Config:



Public Attributes

- hal::AdcDma::Config adc
- hal::Pwm::Config led_pwm
- float max_distance

7.21.1 Detailed Description

template<uint8_t num_of_sensors>
struct proxy::DistanceSensors< num_of_sensors >::Config

Configuration structure for distance sensors.

7.21.2 Member Data Documentation

7.21.2.1 adc

```
template<uint8_t num_of_sensors>
hal::AdcDma::Config proxy::DistanceSensors< num_of_sensors >::Config::adc
```

7.21.2.2 led_pwm

```
template<uint8_t num_of_sensors>
hal::Pwm::Config proxy::DistanceSensors< num_of_sensors >::Config::led_pwm
```

7.21.2.3 max_distance

```
template<uint8_t num_of_sensors>
float proxy::DistanceSensors< num_of_sensors >::Config::max_distance
```

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

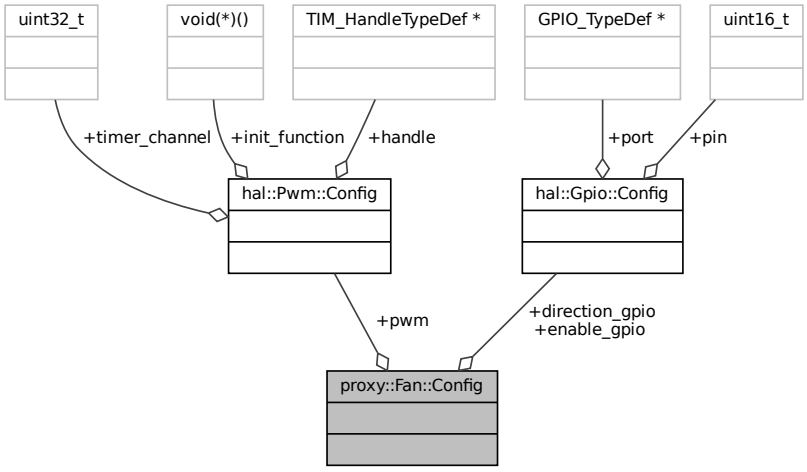
- inc/proxy/distance_sensors.hpp

7.22 proxy::Fan::Config Struct Reference

Configuration structure for the fan.

`#include <fan.hpp>`

Collaboration diagram for proxy::Fan::Config:



Public Attributes

- hal::Pwm::Config pwm
- hal::Gpio::Config direction_gpio
- hal::Gpio::Config enable_gpio

7.22.1 Detailed Description

Configuration structure for the fan.

7.22.2 Member Data Documentation

7.22.2.1 direction_gpio

```
hal::Gpio::Config proxy::Fan::Config::direction_gpio
```

7.22.2.2 enable_gpio

```
hal::Gpio::Config proxy::Fan::Config::enable_gpio
```

7.22.2.3 pwm

```
hal::Pwm::Config proxy::Fan::Config::pwm
```

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

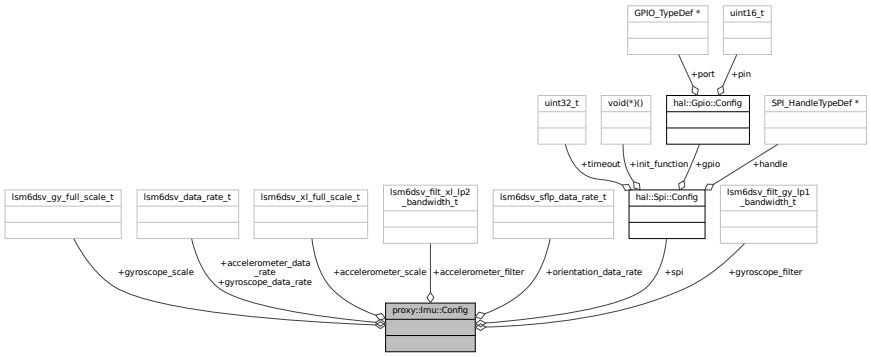
- inc/proxy/fan.hpp

7.23 proxy::Imu::Config Struct Reference

IMU configuration struct.

`#include <imu.hpp>`

Collaboration diagram for proxy::Imu::Config:



Public Attributes

- hal::Spi::Config spi
- lsm6dsv_data_rate_t gyroscope_data_rate
- lsm6dsv_data_rate_t accelerometer_data_rate
- lsm6dsv_sflp_data_rate_t orientation_data_rate
- lsm6dsv_gy_full_scale_t gyroscope_scale
- lsm6dsv_xl_full_scale_t accelerometer_scale
- lsm6dsv_filt_gy_lp1_bandwidth_t gyroscope_filter
- lsm6dsv_filt_xl_lp2_bandwidth_t accelerometer_filter

7.23.1 Detailed Description

IMU configuration struct.

7.23.2 Member Data Documentation

7.23.2.1 accelerometer_data_rate

```
lsm6dsv_data_rate_t proxy::Imu::Config::accelerometer_data_rate
```

7.23.2.2 accelerometer_filter

```
lsm6dsv_filt_xl_lp2_bandwidth_t proxy::Imu::Config::accelerometer_filter
```

7.23.2.3 accelerometer_scale

```
lsm6dsv_xl_full_scale_t proxy::Imu::Config::accelerometer_scale
```

7.23.2.4 gyroscope_data_rate

```
lsm6dsv_data_rate_t proxy::Imu::Config::gyroscope_data_rate
```

7.23.2.5 gyroscope_filter

```
lsm6dsv_filt_gy_lp1_bandwidth_t proxy::Imu::Config::gyroscope_filter
```

7.23.2.6 gyroscope_scale

```
lsm6dsv_gy_full_scale_t proxy::Imu::Config::gyroscope_scale
```

7.23.2.7 orientation_data_rate

```
lsm6dsv_sflp_data_rate_t proxy::Imu::Config::orientation_data_rate
```

7.23.2.8 spi

```
hal::Spi::Config proxy::Imu::Config::spi
```

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

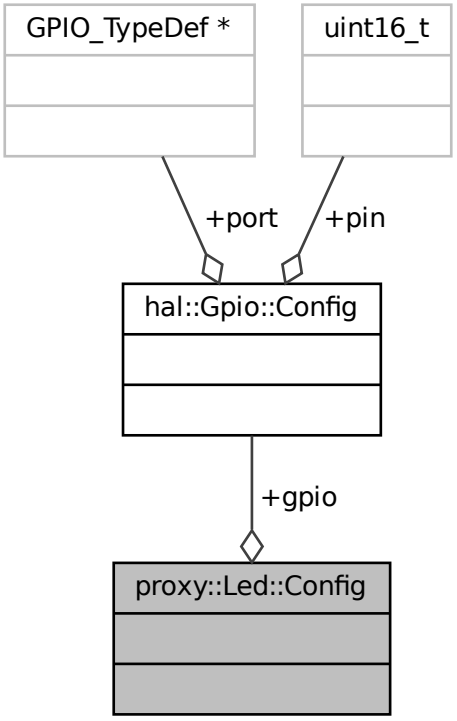
- inc/proxy/imu.hpp

7.24 proxy::Led::Config Struct Reference

Configuration structure for LED.

```
#include <led.hpp>
```

Collaboration diagram for proxy::Led::Config:



Public Attributes

- hal::Gpio::Config gpio

7.24.1 Detailed Description

Configuration structure for LED.

7.24.2 Member Data Documentation

7.24.2.1 gpio

`hal::Gpio::Config proxy::Led::Config::gpio`

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

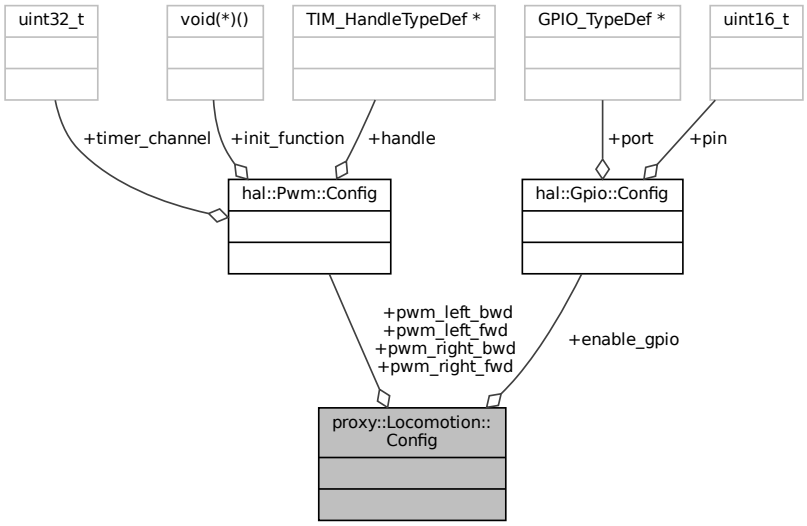
- inc/proxy/led.hpp

7.25 proxy::Locomotion::Config Struct Reference

Configuration structure for the locomotion.

`#include <locomotion.hpp>`

Collaboration diagram for proxy::Locomotion::Config:



Public Attributes

- hal::Pwm::Config pwm_left_fwd
- hal::Pwm::Config pwm_left_bwd
- hal::Pwm::Config pwm_right_fwd
- hal::Pwm::Config pwm_right_bwd
- hal::Gpio::Config enable_gpio

7.25.1 Detailed Description

Configuration structure for the locomotion.

7.25.2 Member Data Documentation

7.25.2.1 enable_gpio

`hal::Gpio::Config proxy::Locomotion::Config::enable_gpio`

7.25.2.2 pwm_left_bwd

```
hal::Pwm::Config proxy::Locomotion::Config::pwm_left_bwd
```

7.25.2.3 pwm_left_fwd

```
hal::Pwm::Config proxy::Locomotion::Config::pwm_left_fwd
```

7.25.2.4 pwm_right_bwd

```
hal::Pwm::Config proxy::Locomotion::Config::pwm_right_bwd
```

7.25.2.5 pwm_right_fwd

```
hal::Pwm::Config proxy::Locomotion::Config::pwm_right_fwd
```

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

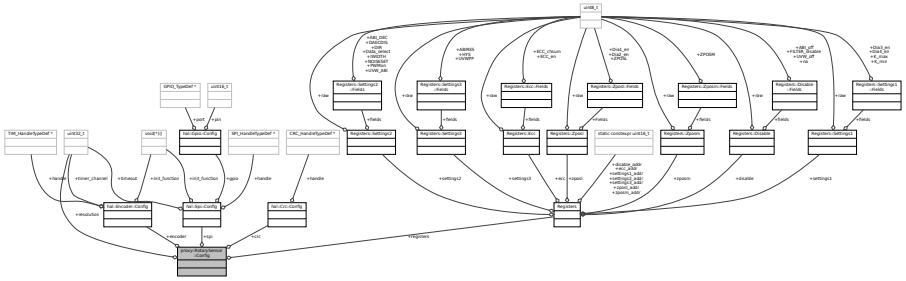
- inc/proxy/locomotion.hpp

7.26 proxy::RotarySensor::Config Struct Reference

Rotary sensor configuration struct.

```
#include <rotary_sensor.hpp>
```

Collaboration diagram for proxy::RotarySensor::Config:



Public Attributes

- hal::Spi::Config spi
- hal::Encoder::Config encoder
- hal::Crc::Config crc
- uint32_t resolution
- Registers registers

7.26.1 Detailed Description

Rotary sensor configuration struct.

7.26.2 Member Data Documentation

7.26.2.1 crc

```
hal::Crc::Config proxy::RotarySensor::Config::crc
```

7.26.2.2 encoder

```
hal::Encoder::Config proxy::RotarySensor::Config::encoder
```

7.26.2.3 registers

```
Registers proxy::RotarySensor::Config::registers
```

7.26.2.4 resolution

```
uint32_t proxy::RotarySensor::Config::resolution
```

7.26.2.5 spi

```
hal::Spi::Config proxy::RotarySensor::Config::spi
```

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

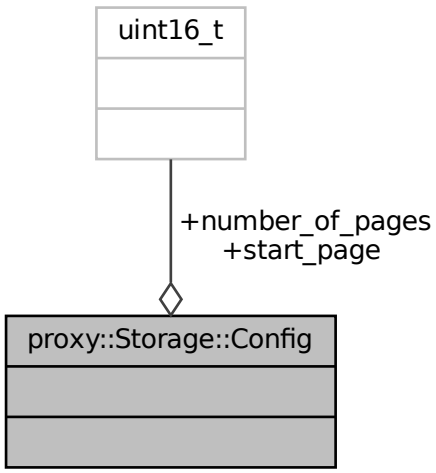
- inc/proxy/rotary_sensor.hpp

7.27 proxy::Storage::Config Struct Reference

Configuration structure for the storage.

`#include <storage.hpp>`

Collaboration diagram for proxy::Storage::Config:



Public Attributes

- uint16_t start_page
- uint16_t number_of_pages

7.27.1 Detailed Description

Configuration structure for the storage.

7.27.2 Member Data Documentation

7.27.2.1 number_of_pages

`uint16_t proxy::Storage::Config::number_of_pages`

7.27.2.2 start_page

```
uint16_t proxy::Storage::Config::start_page
```

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

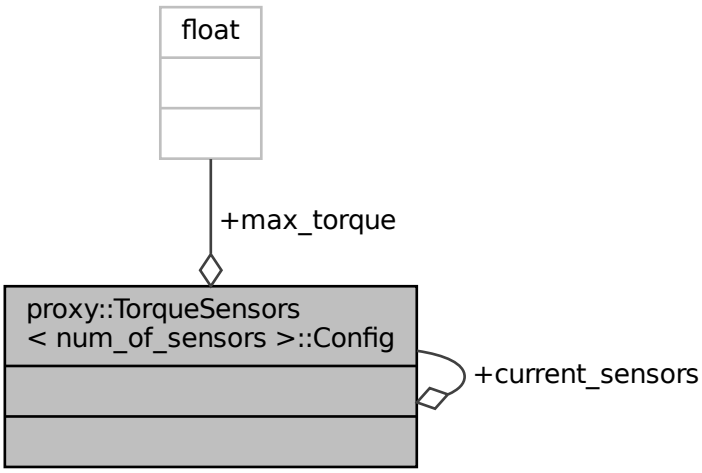
- inc/proxy/storage.hpp

7.28 proxy::TorqueSensors< num_of_sensors >::Config Struct Reference

Configuration structure for torque sensors.

```
#include <torque_sensors.hpp>
```

Collaboration diagram for proxy::TorqueSensors< num_of_sensors >::Config:



Public Attributes

- CurrentSensors< num_of_sensors >::Config current_sensors
- float max_torque

7.28.1 Detailed Description

template<uint8_t num_of_sensors>
struct proxy::TorqueSensors< num_of_sensors >::Config

Configuration structure for torque sensors.

7.28.2 Member Data Documentation

7.28.2.1 current_sensors

```
template<uint8_t num_of_sensors>
CurrentSensors<num_of_sensors>::Config proxy::TorqueSensors< num_of_sensors >::Config::current↩
_sensors
```

7.28.2.2 max_torque

```
template<uint8_t num_of_sensors>
float proxy::TorqueSensors< num_of_sensors >::Config::max_torque
```

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

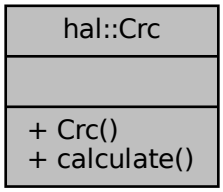
- inc/proxy/torque_sensors.hpp

7.29 hal::Crc Class Reference

Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.

```
#include <crc.hpp>
```

Collaboration diagram for hal::Crc:



Classes

- struct Config

  CRC configuration struct.

Public Member Functions

- Crc (const Config &config)

  Construct a new Crc object.
- uint32_t calculate (uint32_t data[ ], uint32_t size)

  Calculate the CRC value.

7.29.1 Detailed Description

Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.

7.29.2 Constructor & Destructor Documentation

7.29.2.1 Crc()

```
hal::Crc::Crc (
            const Config & config )  [explicit]
```

Construct a new Crc object.

Parameters

config	Configuration for the CRC

7.29.3 Member Function Documentation

7.29.3.1 calculate()

```
uint32_t hal::Crc::calculate (
            uint32_t data[],
            uint32_t size )
```

Calculate the CRC value.

Parameters

data	Data to calculate the CRC
size	Size of the buffer

Returns

uint32_t CRC value

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

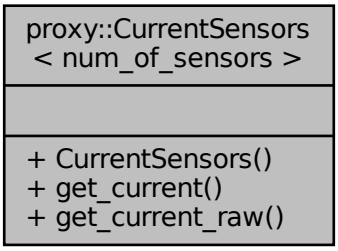
- inc/hal/crc.hpp
- src/hal/crc.cpp

7.30 proxy::CurrentSensors< num_of_sensors > Class Template Reference

Class for controlling CurrentSensors.

`#include <current_sensors.hpp>`

Collaboration diagram for proxy::CurrentSensors< num_of_sensors >:



Classes

- struct Config

  Configuration structure for current sensors.

Public Member Functions

- CurrentSensors (const Config &config)

  Constructor for the CurrentSensors class.
- float get_current (uint8_t sensor_index) const

  Get the current from the sensor.
- uint32_t get_current_raw (uint8_t sensor_index) const

  Get the raw reading from the current sensor.

7.30.1 Detailed Description

template<uint8_t num_of_sensors>
class proxy::CurrentSensors< num_of_sensors >

Class for controlling CurrentSensors.

7.30.2 Constructor & Destructor Documentation

7.30.2.1 CurrentSensors()

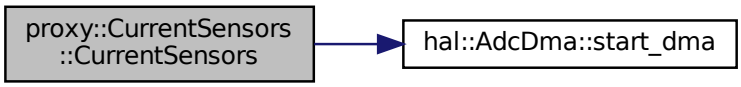
```
template<uint8_t num_of_sensors>
proxy::CurrentSensors< num_of_sensors >::CurrentSensors (
            const Config & config )  [explicit]
```

Constructor for the CurrentSensors class.

Parameters

config	Configuration for the current sensors

Here is the call graph for this function:



7.30.3 Member Function Documentation

7.30.3.1 get_current()

```
template<uint8_t num_of_sensors>
float proxy::CurrentSensors< num_of_sensors >::get_current (
            uint8_t sensor_index ) const
```

Get the current from the sensor.

Parameters

sensor_index	Index of the sensor

Returns

float Current reading from the sensor in amps

7.30.3.2 get_current_raw()

```
template<uint8_t num_of_sensors>
uint32_t proxy::CurrentSensors< num_of_sensors >::get_current_raw (
            uint8_t sensor_index ) const
```

Get the raw reading from the current sensor.

Parameters

sensor_index	Index of the sensor

Returns

uint16_t Current reading from the sensor

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

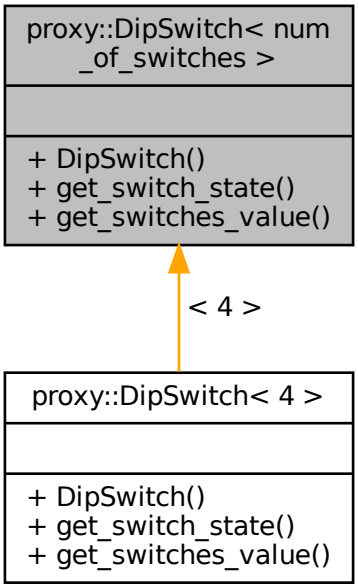
- inc/proxy/current_sensors.hpp
- src/proxy/current_sensors.cpp

7.31 proxy::DipSwitch< num_of_switches > Class Template Reference

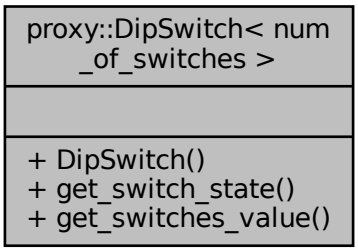
Class for controlling a dip switch.

```
#include <dip_switch.hpp>
```

Inheritance diagram for proxy::DipSwitch< num_of_switches >:



Collaboration diagram for proxy::DipSwitch< num_of_switches >:



Classes

- struct Config

  Configuration struct for DipSwitch.

Public Member Functions

- DipSwitch (const Config &config)

Construct a new Dip Switch object.

- bool get_switch_state (uint8_t switch_index) const

  Get the state of a switch.

- uint8_t get_switches_value () const

  Get the value of all switches.

7.31.1 Detailed Description

template<uint8_t num_of_switches>
class proxy::DipSwitch< num_of_switches >

Class for controlling a dip switch.

7.31.2 Constructor & Destructor Documentation

7.31.2.1 DipSwitch()

```
template<uint8_t num_of_sensors>
proxy::DipSwitch< num_of_sensors >::DipSwitch (
            const Config & config )  [explicit]
```

Construct a new Dip Switch object.

Parameters

config	Configuration struct for DipSwitch

7.31.3 Member Function Documentation

7.31.3.1 get_switch_state()

```
template<uint8_t num_of_sensors>
bool proxy::DipSwitch< num_of_sensors >::get_switch_state (
            uint8_t switch_index ) const
```

Get the state of a switch.

Parameters

switch_index	Index of the switch

Returns

bool True if the switch is on, false otherwise

7.31.3.2 get_switches_value()

```
template<uint8_t num_of_sensors>
uint8_t proxy::DipSwitch< num_of_sensors >::get_switches_value
```

Get the value of all switches.

Returns

uint8_t Value of all switches

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

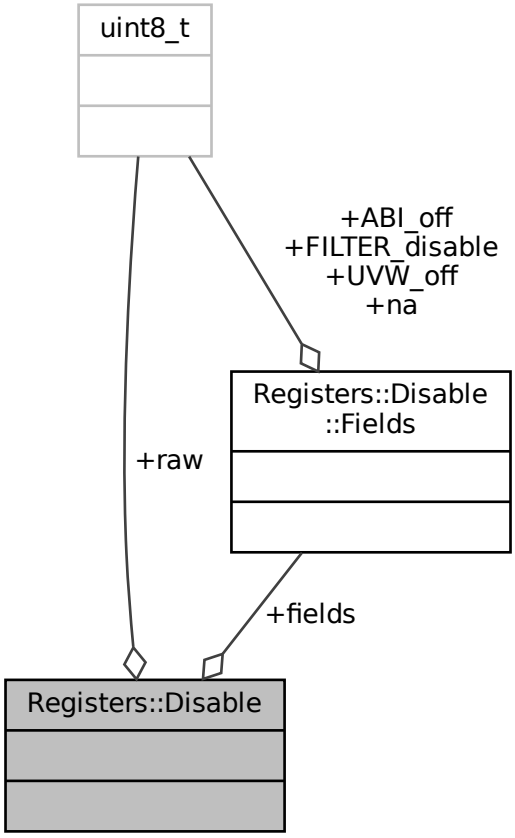
- inc/proxy/dip_switch.hpp
- src/proxy/dip_switch.cpp

7.32 Registers::Disable Union Reference

Registers union types definition.

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Disable:



Classes

- struct Fields

Public Attributes

- Fields fields
- uint8_t raw

7.32.1 Detailed Description

Registers union types definition.

7.32.2 Member Data Documentation

7.32.2.1 fields

```
Fields Registers::Disable::fields
```

7.32.2.2 raw

```
uint8_t Registers::Disable::raw
```

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

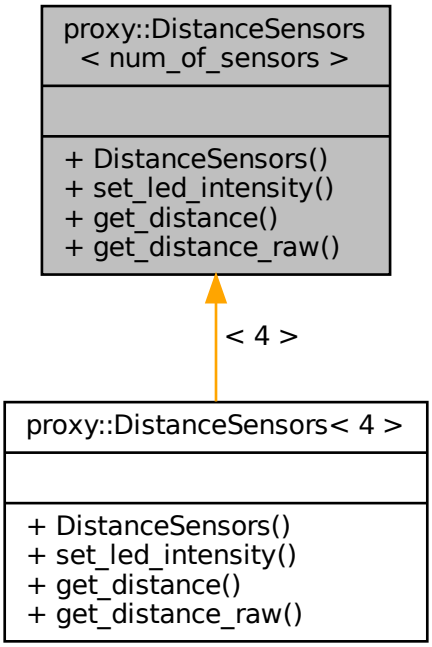
- inc/proxy/rotary_sensor_reg.hpp

7.33 proxy::DistanceSensors< num_of_sensors > Class Template Reference

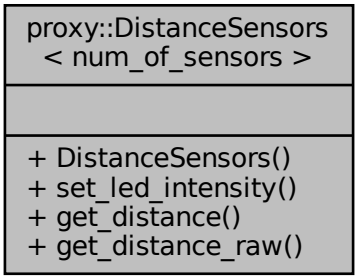
Class for controlling DistanceSensors.

```
#include <distance_sensors.hpp>
```

Inheritance diagram for proxy::DistanceSensors< num_of_sensors >:



Collaboration diagram for proxy::DistanceSensors< num_of_sensors >:



Classes

- struct Config

  Configuration structure for distance sensors.

Public Member Functions

- DistanceSensors (const Config &config)

  Constructor for the DistanceSensors class.
- void set_led_intensity (float intensity)

  Set the distance sensors led intensity.
- float get_distance (uint8_t sensor_index) const

  Get the distance from a sensor.
- uint32_t get_distance_raw (uint8_t sensor_index) const

  Get the distance from a sensor.

7.33.1 Detailed Description

template<uint8_t num_of_sensors>
class proxy::DistanceSensors< num_of_sensors >

Class for controlling DistanceSensors.

7.33.2 Constructor & Destructor Documentation

7.33.2.1 DistanceSensors()

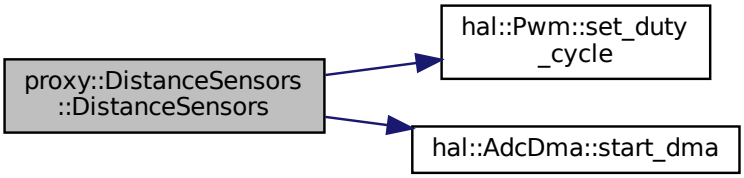
```
template<uint8_t num_of_sensors>
proxy::DistanceSensors< num_of_sensors >::DistanceSensors (
            const Config & config )  [explicit]
```

Constructor for the DistanceSensors class.

Parameters

config	Configuration for the distance sensors

Here is the call graph for this function:



7.33.3 Member Function Documentation

7.33.3.1 get_distance()

```
template<uint8_t num_of_sensors>
float proxy::DistanceSensors< num_of_sensors >::get_distance (
            uint8_t sensor_index ) const
```

Get the distance from a sensor.

Parameters

sensor_index	Index of the sensor

Returns

float Distance reading from the sensors

7.33.3.2 get_distance_raw()

```
template<uint8_t num_of_sensors>
uint32_t proxy::DistanceSensors< num_of_sensors >::get_distance_raw (
            uint8_t sensor_index ) const
```

Get the distance from a sensor.

Parameters

sensor_index	Index of the sensor

Returns

uint32_t Raw reading from the distance sensor

7.33.3.3 set_led_intensity()

```
template<uint8_t num_of_sensors>
void proxy::DistanceSensors< num_of_sensors >::set_led_intensity (
            float intensity )
```

Set the distance sensors led intensity.

Parameters

intensity	Intensity percentage of the infrared LED

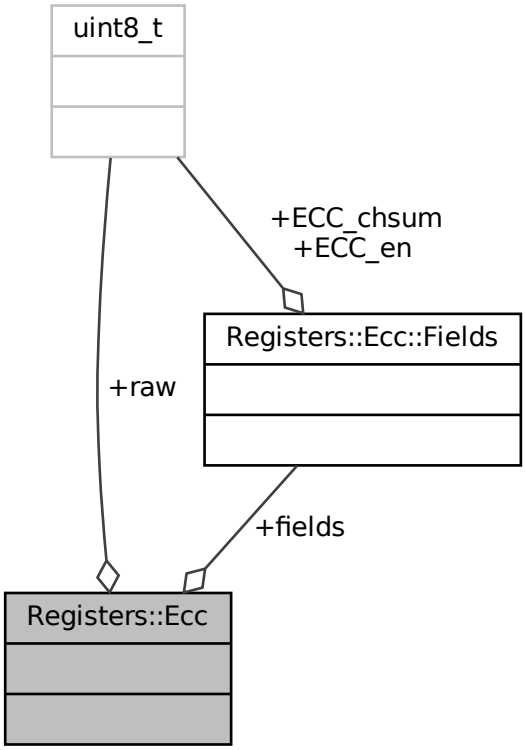
The documentation for this class was generated from the following files:

- inc/proxy/distance_sensors.hpp
- src/proxy/distance_sensors.cpp

7.34 Registers::Ecc Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Ecc:



Classes

- struct Fields

Public Attributes

- Fields fields
- uint8_t raw

7.34.1 Member Data Documentation

7.34.1.1 fields

```
Fields Registers::Ecc::fields
```

7.34.1.2 raw

```
uint8_t Registers::Ecc::raw
```

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

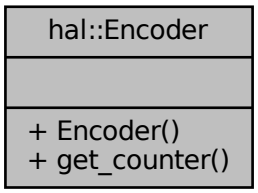
- inc/proxy/rotary_sensor_reg.hpp

7.35 hal::Encoder Class Reference

Class to handle encoder peripheral on STM32 microcontrollers.

`#include <encoder.hpp>`

Collaboration diagram for hal::Encoder:



Classes

- struct Config

  Encoder configuration struct.

Public Member Functions

- Encoder (const Config &config)

  Construct a new Encoder object.
- int32_t get_counter () const

  Get the counter value.

7.35.1 Detailed Description

Class to handle encoder peripheral on STM32 microcontrollers.

7.35.2 Constructor & Destructor Documentation

7.35.2.1 Encoder()

```
hal::Encoder::Encoder (
            const Config & config )  [explicit]
```

Construct a new Encoder object.

Parameters

config	Configuration for the encoder

7.35.3 Member Function Documentation

7.35.3.1 get_counter()

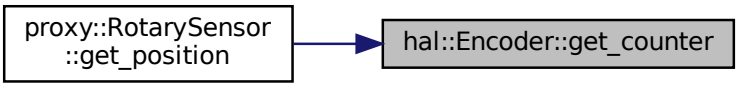
```
int32_t hal::Encoder::get_counter ( ) const
```

Get the counter value.

Returns

int32_t Current value of the counter

Here is the caller graph for this function:



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

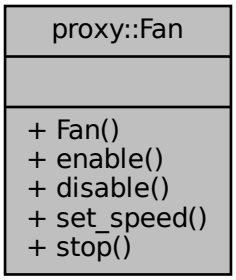
- inc/hal/encoder.hpp
- src/hal/encoder.cpp

7.36 proxy::Fan Class Reference

Class for controlling the fan driver.

```
#include <fan.hpp>
```

Collaboration diagram for proxy::Fan:



Classes

- struct Config

  Configuration structure for the fan.

Public Types

- enum RotationDirection { FORWARD , BACKWARD }

  Enum for rotation direction.

Public Member Functions

- Fan (const Config &config)

  Construct a new fan object.
- void enable ()

  Enable the fan.
- void disable ()

  Disable the fan.
- void set_speed (float speed)

  Set the speed of the fans.
- void stop ()

  Stop the fan.

7.36.1 Detailed Description

Class for controlling the fan driver.

7.36.2 Member Enumeration Documentation

7.36.2.1 RotationDirection

`enum proxy::Fan::RotationDirection`

Enum for rotation direction.

Enumerator

FORWARD	
BACKWARD	

7.36.3 Constructor & Destructor Documentation

7.36.3.1 Fan()

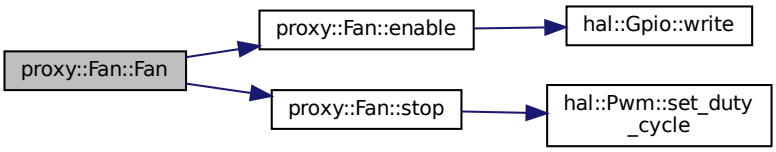
```
proxy::Fan::Fan (
            const Config & config )  [explicit]
```

Construct a new fan object.

Parameters

config	Configuration for the fan driver

Here is the call graph for this function:



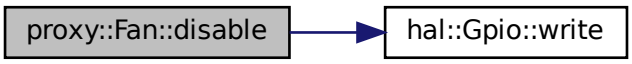
7.36.4 Member Function Documentation

7.36.4.1 disable()

```
void proxy::Fan::disable ( )
```

Disable the fan.

Here is the call graph for this function:



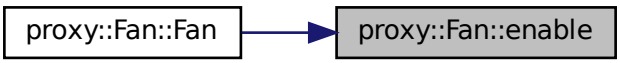
7.36.4.2 enable()

```
void proxy::Fan::enable ( )
```

Enable the fan.

Here is the call graph for this function:

Here is the caller graph for this function:



7.36.4.3 set_speed()

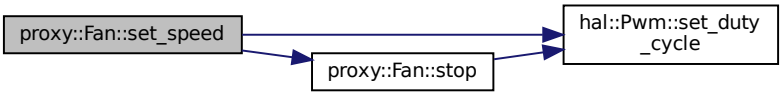
```
void proxy::Fan::set_speed (
            float speed )
```

Set the speed of the fans.

Parameters

speed	Speed percentage of the fan

Here is the call graph for this function:

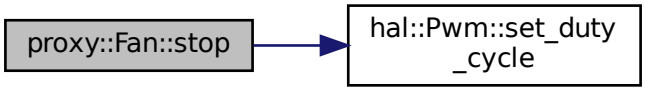


7.36.4.4 stop()

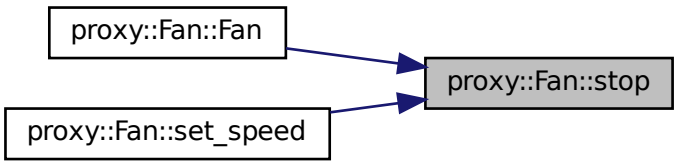
```
void proxy::Fan::stop ( )
```

Stop the fan.

Here is the call graph for this function:



Here is the caller graph for this function:



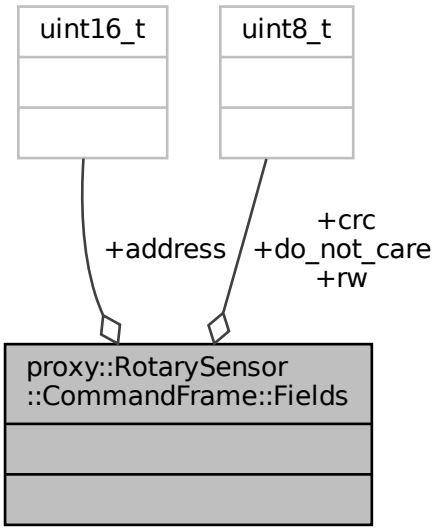
The documentation for this class was generated from the following files:

- inc/proxy/fan.hpp
- src/proxy/fan.cpp

7.37 proxy::RotarySensor::CommandFrame::Fields Struct Reference

`#include <rotary_sensor.hpp>`

Collaboration diagram for proxy::RotarySensor::CommandFrame::Fields:



Public Attributes

- uint8_t do_not_care: 1
- uint8_t rw: 1
- uint16_t address: 14
- uint8_t crc: 8

7.37.1 Member Data Documentation

7.37.1.1 address

`uint16_t proxy::RotarySensor::CommandFrame::Fields::address`

7.37.1.2 crc

`uint8_t proxy::RotarySensor::CommandFrame::Fields::crc`

7.37.1.3 do_not_care

```
uint8_t proxy::RotarySensor::CommandFrame::Fields::do_not_care
```

7.37.1.4 rw

```
uint8_t proxy::RotarySensor::CommandFrame::Fields::rw
```

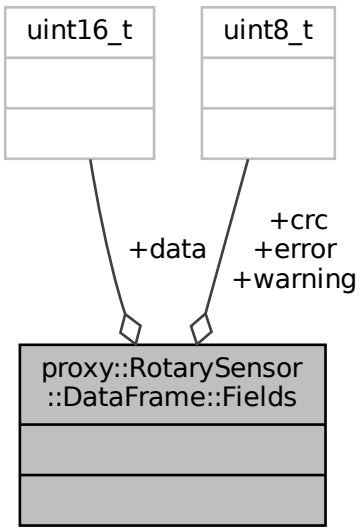
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor.hpp

7.38 proxy::RotarySensor::DataFrame::Fields Struct Reference

```
#include <rotary_sensor.hpp>
```

Collaboration diagram for proxy::RotarySensor::DataFrame::Fields:



Public Attributes

- uint8_t warning: 1
- uint8_t error: 1
- uint16_t data: 14
- uint8_t crc: 8

7.38.1 Member Data Documentation

7.38.1.1 crc

```
uint8_t proxy::RotarySensor::DataFrame::Fields::crc
```

7.38.1.2 data

```
uint16_t proxy::RotarySensor::DataFrame::Fields::data
```

7.38.1.3 error

```
uint8_t proxy::RotarySensor::DataFrame::Fields::error
```

7.38.1.4 warning

```
uint8_t proxy::RotarySensor::DataFrame::Fields::warning
```

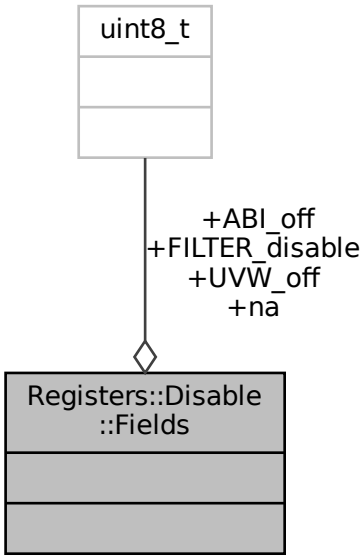
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor.hpp

7.39 Registers::Disable::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Disable::Fields:



Public Attributes

- uint8_t UVW_off: 1
- uint8_t ABI_off: 1
- uint8_t na: 4
- uint8_t FILTER_disable: 1

7.39.1 Member Data Documentation

7.39.1.1 ABI_off

```
uint8_t Registers::Disable::Fields::ABI_off
```

7.39.1.2 FILTER_disable

```
uint8_t Registers::Disable::Fields::FILTER_disable
```

7.39.1.3 na

```
uint8_t Registers::Disable::Fields::na
```

7.39.1.4 UVW_off

```
uint8_t Registers::Disable::Fields::UVW_off
```

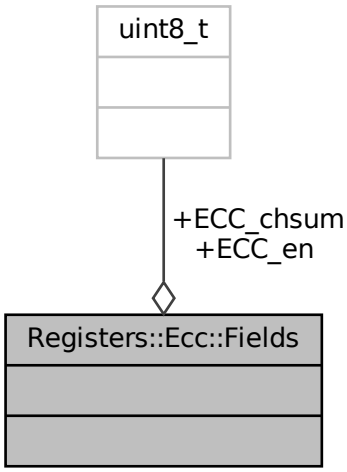
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.40 Registers::Ecc::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Ecc::Fields:



Public Attributes

- uint8_t ECC_chsum: 7
- uint8_t ECC_en: 1

7.40.1 Member Data Documentation

7.40.1.1 ECC_chsum

```
uint8_t Registers::Ecc::Fields::ECC_chsum
```

7.40.1.2 ECC_en

```
uint8_t Registers::Ecc::Fields::ECC_en
```

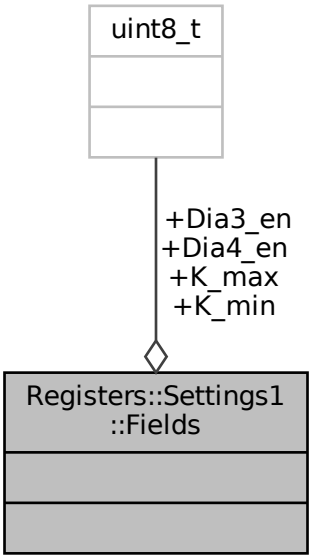
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.41 Registers::Settings1::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings1::Fields:



Public Attributes

- uint8_t K_max: 3
- uint8_t K_min: 3
- uint8_t Dia3_en: 1
- uint8_t Dia4_en: 1

7.41.1 Member Data Documentation

7.41.1.1 Dia3_en

```
uint8_t Registers::Settings1::Fields::Dia3_en
```

7.41.1.2 Dia4_en

```
uint8_t Registers::Settings1::Fields::Dia4_en
```

7.41.1.3 K_max

```
uint8_t Registers::Settings1::Fields::K_max
```

7.41.1.4 K_min

```
uint8_t Registers::Settings1::Fields::K_min
```

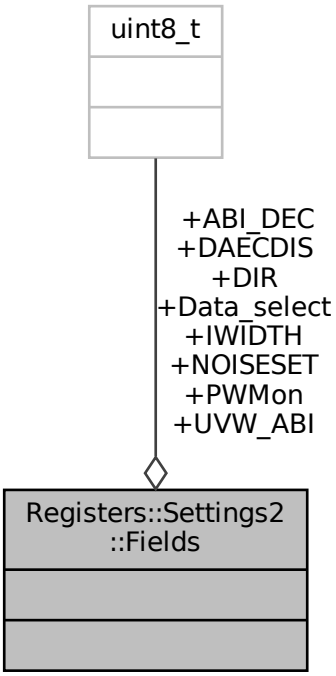
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.42 Registers::Settings2::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings2::Fields:



Public Attributes

- uint8_t IWIDTH: 1
- uint8_t NOISESET: 1
- uint8_t DIR: 1
- uint8_t UVW_ABI: 1
- uint8_t DAECDIS: 1
- uint8_t ABI_DEC: 1
- uint8_t Data_select: 1
- uint8_t PWMon: 1

7.42.1 Member Data Documentation

7.42.1.1 ABI_DEC

```
uint8_t Registers::Settings2::Fields::ABI_DEC
```

7.42.1.2 DAECDIS

```
uint8_t Registers::Settings2::Fields::DAECDIS
```

7.42.1.3 Data_select

```
uint8_t Registers::Settings2::Fields::Data_select
```

7.42.1.4 DIR

```
uint8_t Registers::Settings2::Fields::DIR
```

7.42.1.5 IWIDTH

```
uint8_t Registers::Settings2::Fields::IWIDTH
```

7.42.1.6 NOISESET

```
uint8_t Registers::Settings2::Fields::NOISESET
```

7.42.1.7 PWMon

```
uint8_t Registers::Settings2::Fields::PWMon
```

7.42.1.8 UVW_ABI

```
uint8_t Registers::Settings2::Fields::UVW_ABI
```

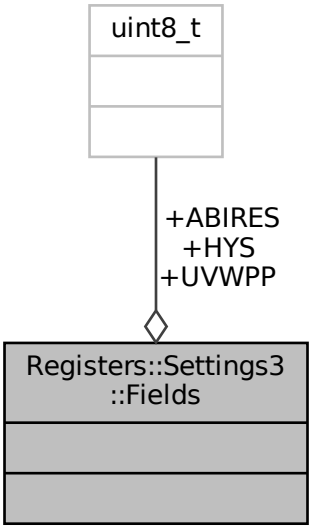
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.43 Registers::Settings3::Fields Struct Reference

`#include <rotary_sensor_reg.hpp>`

Collaboration diagram for Registers::Settings3::Fields:



Public Attributes

- uint8_t UVWPP: 3
- uint8_t HYS: 2
- uint8_t ABIRES: 3

7.43.1 Member Data Documentation

7.43.1.1 ABIRES

`uint8_t Registers::Settings3::Fields::ABIRES`

7.43.1.2 HYS

`uint8_t Registers::Settings3::Fields::HYS`

7.43.1.3 UVWPP

```
uint8_t Registers::Settings3::Fields::UVWPP
```

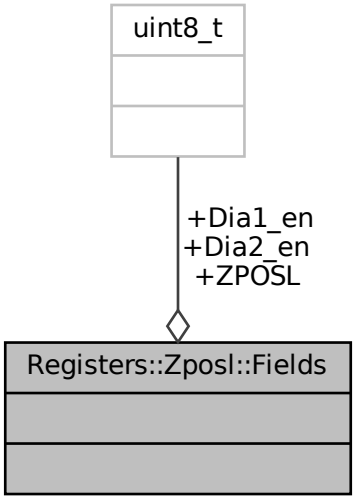
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.44 Registers::Zposl::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposl::Fields:



Public Attributes

- uint8_t ZPOSL: 6
- uint8_t Dia1_en: 1
- uint8_t Dia2_en: 1

7.44.1 Member Data Documentation

7.44.1.1 Dia1_en

```
uint8_t Registers::Zposl::Fields::Dia1_en
```

7.44.1.2 Dia2_en

```
uint8_t Registers::Zposl::Fields::Dia2_en
```

7.44.1.3 ZPOSL

```
uint8_t Registers::Zposl::Fields::ZPOSL
```

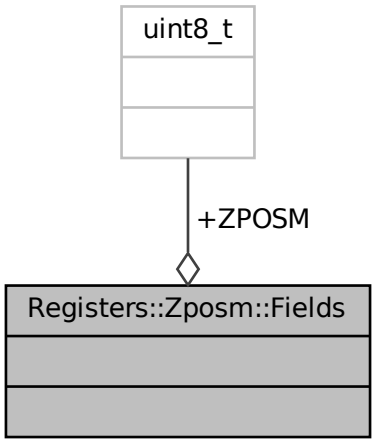
The documentation for this struct was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.45 Registers::Zposm::Fields Struct Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposm::Fields:



Public Attributes

- uint8_t ZPOSM: 8

7.45.1 Member Data Documentation

7.45.1.1 ZPOSM

`uint8_t Registers::Zposm::Fields::ZPOSM`

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

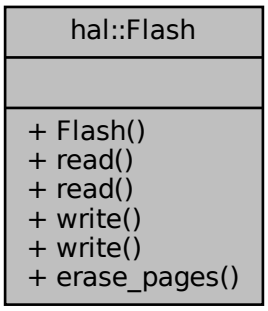
- inc/proxy/rotary_sensor_reg.hpp

7.46 hal::Flash Class Reference

Class to handle flash memory on STM32 microcontrollers.

`#include <flash.hpp>`

Collaboration diagram for hal::Flash:



Public Member Functions

- Flash ()=delete

  Deleted constructor for static class.

Static Public Member Functions

- static void read (uint32_t address, uint64_t data[ ], uint32_t size=1)

  Read data from flash memory.
- static void read (uint16_t page, uint16_t page_address, uint64_t data[ ], uint32_t size=1)

  Read data from flash memory.
- static void write (uint32_t address, const uint64_t data[ ], uint32_t size=1)

  Write data to flash memory.
- static void write (uint16_t page, uint16_t page_address, const uint64_t data[ ], uint32_t size=1)

  Write data to flash memory.
- static void erase_pages (uint16_t page, uint16_t number_of_pages=1)

  Erase flash memory pages.

7.46.1 Detailed Description

Class to handle flash memory on STM32 microcontrollers.

7.46.2 Constructor & Destructor Documentation

7.46.2.1 Flash()

```
hal::Flash::Flash ( )  [delete]
```

Deleted constructor for static class.

7.46.3 Member Function Documentation

7.46.3.1 erase_pages()

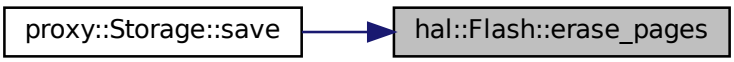
```
void hal::Flash::erase_pages (
            uint16_t page,
            uint16_t number_of_pages = 1 )  [static]
```

Erase flash memory pages.

Parameters

page	first page to erase (counting from the last to the first)
number_of_pages	number of pages to erase

Here is the caller graph for this function:



7.46.3.2 read() [1/2]

```
void hal::Flash::read (
            uint16_t page,
            uint16_t page_address,
            uint64_t data[],
            uint32_t size = 1 )  [static]
```

Read data from flash memory.

Parameters

page	page to read from (counting from the last to the first)
page_address	address inside the page to read from (indexed by double words)
data	pointer to store the data read
size	size in double words of the data to read

Here is the call graph for this function:

7.46.3.3 read() [2/2]

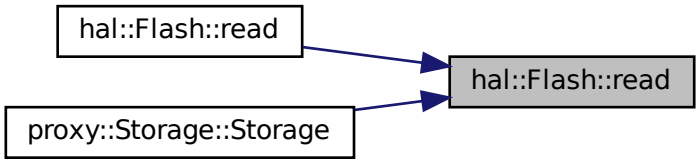
```
void hal::Flash::read (
            uint32_t address,
            uint64_t data[],
            uint32_t size = 1 )  [static]
```

Read data from flash memory.

Parameters

address	address to read from (indexed by double words)
data	pointer to store the data read
size	size in double words of the data to read

Here is the caller graph for this function:



7.46.3.4 write() [1/2]

```
void hal::Flash::write (
            uint16_t page,
            uint16_t page_address,
            const uint64_t data[],
            uint32_t size = 1 )  [static]
```

Write data to flash memory.

Parameters

page	page to write to (counting from the last to the first)
page_address	address inside the page to write to (indexed by double words)
data	pointer to the data to write
size	size in double words of the data to write

Here is the call graph for this function:

7.46.3.5 write() [2/2]

```
void hal::Flash::write (
            uint32_t address,
```

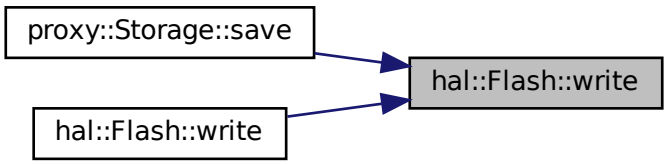
```
const uint64_t data[],
uint32_t size = 1 )  [static]
```

Write data to flash memory.

Parameters

address	address to write to (indexed by double words)
data	pointer to the data to write
size	size in double words of the data to write

Here is the caller graph for this function:



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

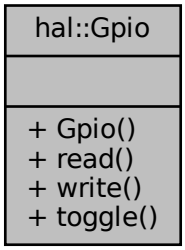
- inc/hal/flash.hpp
- src/hal/flash.cpp

7.47 hal::Gpio Class Reference

Class for controlling GPIO pins on STM32 microcontrollers.

`#include <gpio.hpp>`

Collaboration diagram for hal::Gpio:



Classes

- struct Config

  Configuration structure for GPIO pin.

Public Member Functions

- Gpio (const Config &config)

  Constructor for the Gpio class.
- bool read () const

  Read the current state of the GPIO pin.
- void write (bool state)

  Write a new state to the GPIO pin.
- void toggle ()

  Toggle the state of the GPIO pin.

7.47.1 Detailed Description

Class for controlling GPIO pins on STM32 microcontrollers.

7.47.2 Constructor & Destructor Documentation

7.47.2.1 Gpio()

```
hal::Gpio::Gpio (
            const Config & config )  [explicit]
```

Constructor for the Gpio class.

Parameters

config	Configuration for the GPIO pin

7.47.3 Member Function Documentation

7.47.3.1 read()

```
bool hal::Gpio::read ( ) const
```

Read the current state of the GPIO pin.

Returns

bool The current state of the GPIO pin (true for high, false for low)

7.47.3.2 toggle()

```
void hal::Gpio::toggle ( )
```

Toggle the state of the GPIO pin.

Here is the caller graph for this function:

7.47.3.3 write()

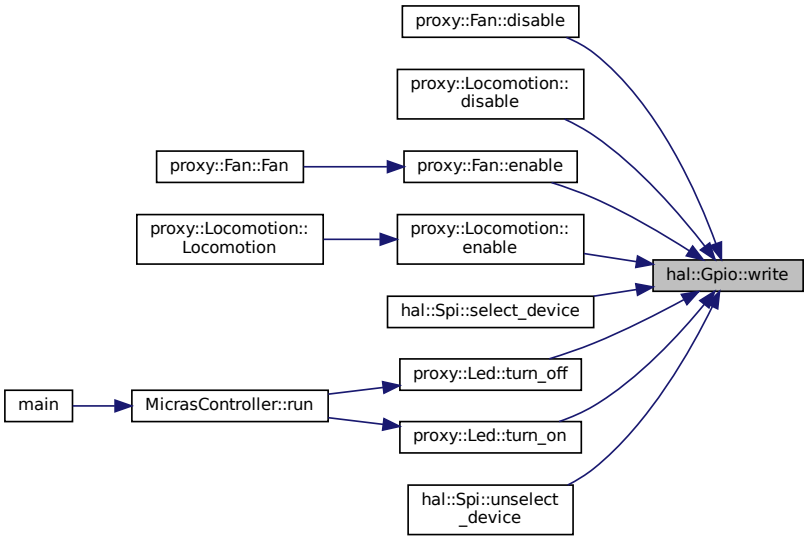
```
void hal::Gpio::write (
            bool state )
```

Write a new state to the GPIO pin.

Parameters

pin_state	The state to be written (true for high, false for low)

Here is the caller graph for this function:



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

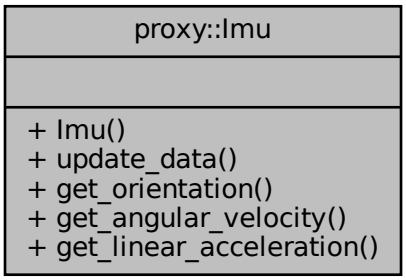
- inc/hal/gpio.hpp
- src/hal/gpio.cpp

7.48 proxy::Imu Class Reference

Class to handle IMU peripheral on STM32 microcontrollers.

`#include <imu.hpp>`

Collaboration diagram for proxy::Imu:



Classes

- struct Config

  IMU configuration struct.

Public Types

- enum Axis { W , X , Y , Z }

Public Member Functions

- Imu (const Config &config)

  Construct a new Imu object.
- void update_data ()

  Update the IMU data.
- float get_orientation (Axis axis) const

  Get the IMU orientation over an axis.
- float get_angular_velocity (Axis axis) const

  Get the IMU angular velocity over an axis.
- float get_linear_acceleration (Axis axis) const

  Get the IMU linear acceleration over an axis.

7.48.1 Detailed Description

Class to handle IMU peripheral on STM32 microcontrollers.

7.48.2 Member Enumeration Documentation

7.48.2.1 Axis

`enum proxy::Imu::Axis`

Enumerator

W	
X	
Y	
Z	

7.48.3 Constructor & Destructor Documentation

7.48.3.1 Imu()

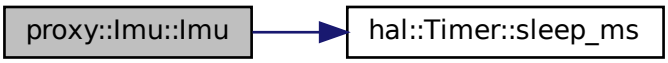
```
proxy::Imu::Imu (
            const Config & config )  [explicit]
```

Construct a new Imu object.

Parameters

config	Configuration for the IMU

Here is the call graph for this function:



7.48.4 Member Function Documentation

7.48.4.1 get_angular_velocity()

```
float proxy::Imu::get_angular_velocity (
            Axis axis ) const
```

Get the IMU angular velocity over an axis.

Parameters

axis	Axis to get the angular velocity from

Returns

float Angular velocity over the desired axis in rad/s

7.48.4.2 get_linear_acceleration()

```
float proxy::Imu::get_linear_acceleration (
            Axis axis ) const
```

Get the IMU linear acceleration over an axis.

Parameters

axis	Axis to get the linear acceleration from

Returns

float Linear acceleration over the desired axis in m/s^2

7.48.4.3 get_orientation()

```
float proxy::Imu::get_orientation (
            Axis axis ) const
```

Get the IMU orientation over an axis.

Todo implement function using sensior fusion

Parameters

axis	Axis to get the orientation from

Returns

float Orientation over the desired axis using quaternions

7.48.4.4 update_data()

```
void proxy::Imu::update_data ( )
```

Update the IMU data.

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

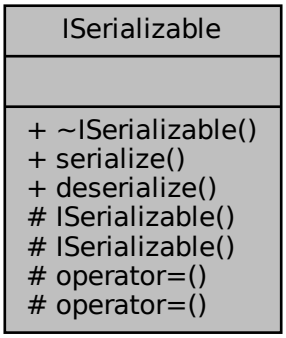
- inc/proxy/imu.hpp
- src/proxy/imu.cpp

7.49 ISerializable Class Reference

Interface class for serializable classes.

`#include <serializable_interface.hpp>`

Collaboration diagram for ISerializable:



Public Member Functions

- virtual ~ISerializable ()=default

  Virtual destructor for the ISerializable class.
- virtual std::vector< uint8_t > serialize () const =0

  Serialize the class instance.
- virtual void deserialize (uint8_t *serial_data, uint16_t size)=0

  Deserialize the class instance.

Protected Member Functions

- ISerializable (const ISerializable &)=default

  Special member functions declared as default.
- ISerializable (ISerializable &&)=default
- ISerializable & operator= (const ISerializable &)=default
- ISerializable & operator= (ISerializable &&)=default

7.49.1 Detailed Description

Interface class for serializable classes.

7.49.2 Constructor & Destructor Documentation

7.49.2.1 ~ISerializable()

```
virtual ISerializable::~ISerializable ( )  [virtual], [default]
```

Virtual destructor for the ISerializable class.

7.49.2.2 ISerializable() [1/2]

```
ISerializable::ISerializable (
            const ISerializable &  )  [protected], [default]
```

Special member functions declared as default.

7.49.2.3 ISerializable() [2/2]

```
ISerializable::ISerializable (
            ISerializable &&  )  [protected], [default]
```

7.49.3 Member Function Documentation

7.49.3.1 deserialize()

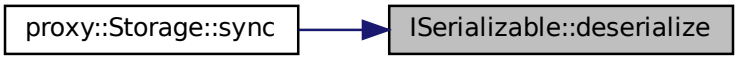
```
virtual void ISerializable::deserialize (
            uint8_t * serial_data,
            uint16_t size )  [pure virtual]
```

Deserialize the class instance.

Parameters

serial_data	Serialized data
size	Size of the serialized data

Here is the caller graph for this function:



7.49.3.2 operator=() [1/2]

```
ISerializable& ISerializable::operator= (
            const ISerializable &  )  [protected], [default]
```

7.49.3.3 operator=() [2/2]

```
ISerializable& ISerializable::operator= (
            ISerializable &&  )  [protected], [default]
```

7.49.3.4 serialize()

```
virtual std::vector<uint8_t> ISerializable::serialize ( ) const  [pure virtual]
```

Serialize the class instance.

Returns

std::vector<uint8_t> Serialized data

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

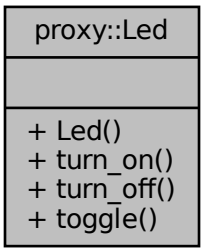
- inc/proxy/serializable_interface.hpp

7.50 proxy::Led Class Reference

Class for controlling an LED.

`#include <led.hpp>`

Collaboration diagram for proxy::Led:



Classes

- struct Config

  Configuration structure for LED.

Public Member Functions

- Led (const Config &config)

  Constructor for the Led class.
- void turn_on ()

  Turn the LED on.
- void turn_off ()

  Turn the LED off.
- void toggle ()

  Toggle the LED.

7.50.1 Detailed Description

Class for controlling an LED.

7.50.2 Constructor & Destructor Documentation

7.50.2.1 Led()

```
proxy::Led::Led (
            const Config & config )  [explicit]
```

Constructor for the Led class.

Parameters

config	Configuration for the LED

7.50.3 Member Function Documentation

7.50.3.1 toggle()

```
void proxy::Led::toggle ( )
```

Toggle the LED.

Here is the call graph for this function:

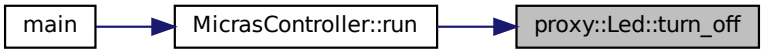
7.50.3.2 turn_off()

```
void proxy::Led::turn_off ( )
```

Turn the LED off.

Here is the call graph for this function:

Here is the caller graph for this function:



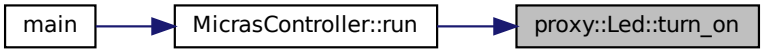
7.50.3.3 turn_on()

```
void proxy::Led::turn_on ( )
```

Turn the LED on.

Here is the call graph for this function:

Here is the caller graph for this function:



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

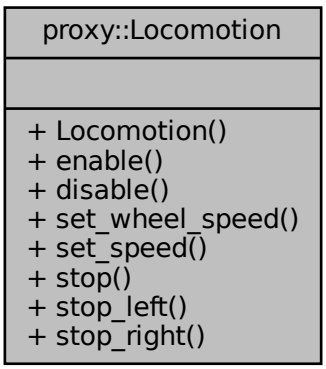
- inc/proxy/led.hpp
- src/proxy/led.cpp

7.51 proxy::Locomotion Class Reference

Class for controlling the locomotion driver.

`#include <locomotion.hpp>`

Collaboration diagram for proxy::Locomotion:



Classes

- struct Config

  Configuration structure for the locomotion.

Public Member Functions

- Locomotion (const Config &config)

  Construct a new locomotion object.
- void enable ()

  Enable the locomotion driver.
- void disable ()

  Disable the locomotion driver.
- void set_wheel_speed (float left_speed, float right_speed)

  Set the speed of the wheels.
- void set_speed (float linear, float angular)

  Set the linear and angular speeds of the robot.
- void stop ()

  Stop the motors.
- void stop_left ()

  Stop the left motor.
- void stop_right ()

  Stop the right motor.

7.51.1 Detailed Description

Class for controlling the locomotion driver.

7.51.2 Constructor & Destructor Documentation

7.51.2.1 Locomotion()

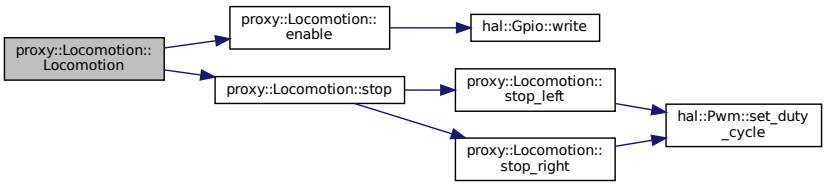
```
proxy::Locomotion::Locomotion (
            const Config & config )  [explicit]
```

Construct a new locomotion object.

Parameters

config	Configuration for the locomotion driver

Here is the call graph for this function:



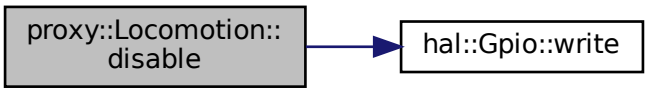
7.51.3 Member Function Documentation

7.51.3.1 disable()

```
void proxy::Locomotion::disable ( )
```

Disable the locomotion driver.

Here is the call graph for this function:

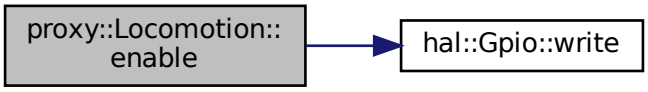


7.51.3.2 enable()

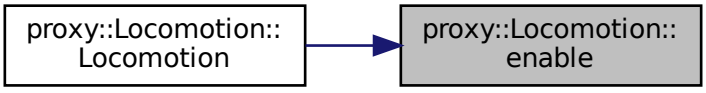
```
void proxy::Locomotion::enable ( )
```

Enable the locomotion driver.

Here is the call graph for this function:



Here is the caller graph for this function:



7.51.3.3 set_speed()

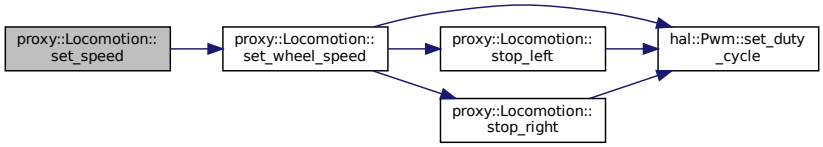
```
void proxy::Locomotion::set_speed (
            float linear,
            float angular )
```

Set the linear and angular speeds of the robot.

Parameters

linear	Linear speed of the robot
angular	Angular speed of the robot

Here is the call graph for this function:



7.51.3.4 set_wheel_speed()

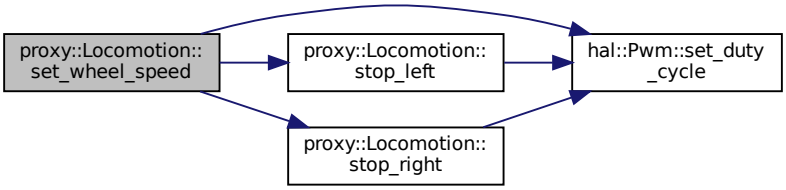
```
void proxy::Locomotion::set_wheel_speed (
            float left_speed,
            float right_speed )
```

Set the speed of the wheels.

Parameters

left_speed	Speed of the left wheels
right_speed	Speed of the right wheels

Here is the call graph for this function:



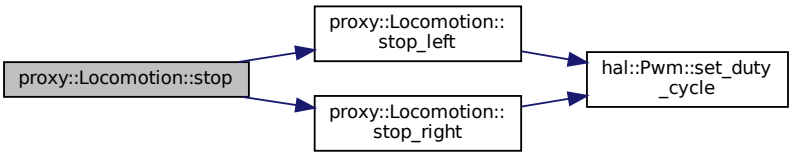
Here is the caller graph for this function:

7.51.3.5 stop()

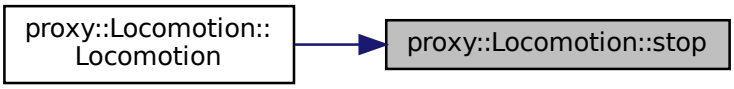
```
void proxy::Locomotion::stop ( )
```

Stop the motors.

Here is the call graph for this function:



Here is the caller graph for this function:

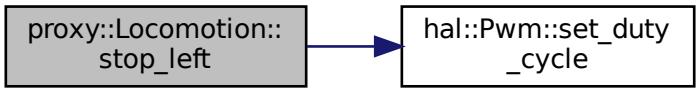


7.51.3.6 stop_left()

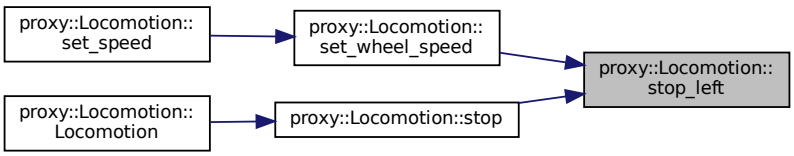
```
void proxy::Locomotion::stop_left ( )
```

Stop the left motor.

Here is the call graph for this function:



Here is the caller graph for this function:

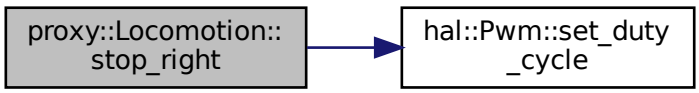


7.51.3.7 stop_right()

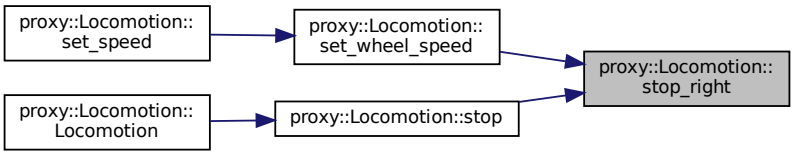
```
void proxy::Locomotion::stop_right ( )
```

Stop the right motor.

Here is the call graph for this function:



Here is the caller graph for this function:



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

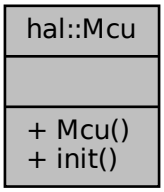
- inc/proxy/locomotion.hpp
- src/proxy/locomotion.cpp

7.52 hal::Mcu Class Reference

Microcontroller unit class.

`#include <mcu.hpp>`

Collaboration diagram for hal::Mcu:



Public Member Functions

- Mcu ()=delete

  Deleted constructor for static class.

Static Public Member Functions

- static void init ()

  Initializes MCU and some peripherals.

7.52.1 Detailed Description

Microcontroller unit class.

7.52.2 Constructor & Destructor Documentation

7.52.2.1 Mcu()

```
hal::Mcu::Mcu ( )  [delete]
```

Deleted constructor for static class.

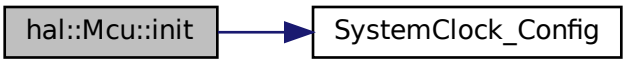
7.52.3 Member Function Documentation

7.52.3.1 init()

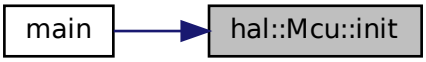
```
void hal::Mcu::init ( )  [static]
```

Initializes MCU and some peripherals.

Here is the call graph for this function:



Here is the caller graph for this function:



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

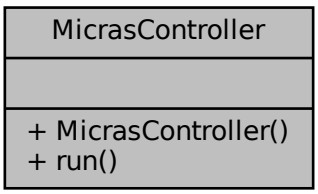
- inc/hal/mcu.hpp
- src/hal/mcu.cpp

7.53 MicrasController Class Reference

Class for controlling the Micras robot.

`#include <micras_controller.hpp>`

Collaboration diagram for MicrasController:



Public Member Functions

- MicrasController ()

  Constructor for the MicrasController class.
- void run ()

  Runs the controller loop once.

7.53.1 Detailed Description

Class for controlling the Micras robot.

7.53.2 Constructor & Destructor Documentation

7.53.2.1 MicrasController()

`MicrasController::MicrasController ( )`

Constructor for the MicrasController class.

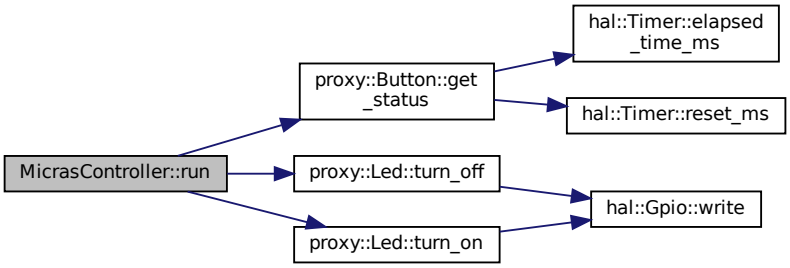
7.53.3 Member Function Documentation

7.53.3.1 run()

```
void MicrasController::run ( )
```

Runs the controller loop once.

Here is the call graph for this function:



Here is the caller graph for this function:

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

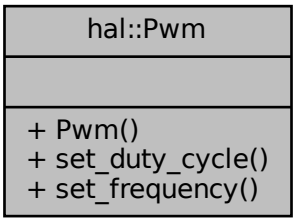
- inc/controller/micras_controller.hpp
- src/controller/micras_controller.cpp

7.54 hal::Pwm Class Reference

Class to handle PWM peripheral on STM32 microcontrollers.

```
#include <pwm.hpp>
```

Collaboration diagram for hal::Pwm:



Classes

- struct Config

  PWM configuration struct.

Public Member Functions

- Pwm (const Config &config)

  Construct a new Pwm object.
- void set_duty_cycle (float duty_cycle)

  Set the PWM duty cycle.
- void set_frequency (uint32_t frequency)

  Set the PWM frequency.

7.54.1 Detailed Description

Class to handle PWM peripheral on STM32 microcontrollers.

7.54.2 Constructor & Destructor Documentation

7.54.2.1 Pwm()

```
hal::Pwm::Pwm (
            const Config & config )  [explicit]
```

Construct a new Pwm object.

Parameters

config	Configuration for the PWM

7.54.3 Member Function Documentation

7.54.3.1 set_duty_cycle()

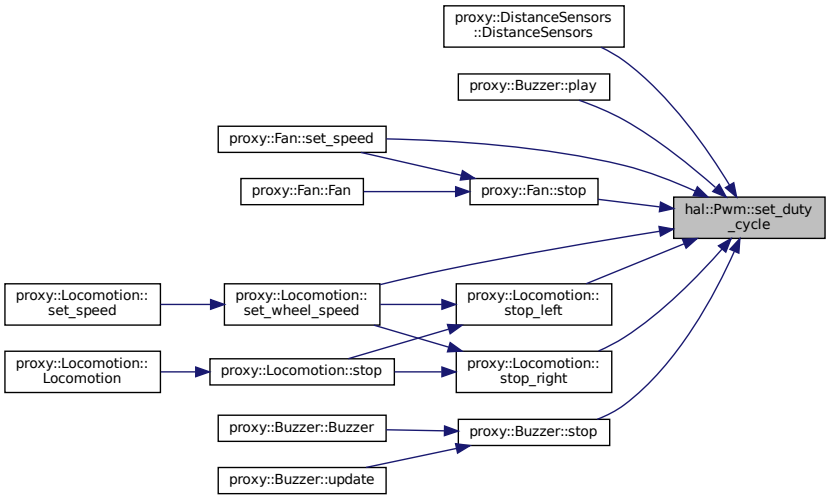
```
void hal::Pwm::set_duty_cycle (
            float duty_cycle )
```

Set the PWM duty cycle.

Parameters

duty_cycle	Duty cycle value

Here is the caller graph for this function:



7.54.3.2 set_frequency()

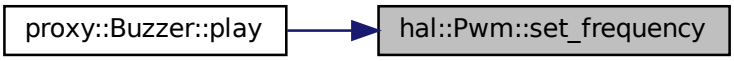
```
void hal::Pwm::set_frequency (
            uint32_t frequency )
```

Set the PWM frequency.

Parameters

frequency	Frequency value in Hz

Here is the caller graph for this function:



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

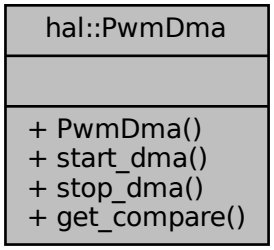
- inc/hal/pwm.hpp
- src/hal/pwm.cpp

7.55 hal::PwmDma Class Reference

Class to handle PWM peripheral on STM32 microcontrollers using DMA.

`#include <pwm_dma.hpp>`

Collaboration diagram for hal::PwmDma:



Classes

- struct Config

  PWM configuration struct.

Public Member Functions

- PwmDma (const Config &config)

  Construct a new PwmDma object.
- void start_dma (uint32_t buffer[ ], uint32_t size)

  Start PWM and DMA transfer.
- void stop_dma ()

  Stop PWM and DMA transfer.
- uint32_t get_compare (float duty_cycle) const

  Get the compare value for ad duty cycle.

7.55.1 Detailed Description

Class to handle PWM peripheral on STM32 microcontrollers using DMA.

7.55.2 Constructor & Destructor Documentation

7.55.2.1 PwmDma()

```
hal::PwmDma::PwmDma (
            const Config & config )  [explicit]
```

Construct a new PwmDma object.

Parameters

config	Configuration for the PWM

7.55.3 Member Function Documentation

7.55.3.1 get_compare()

```
uint32_t hal::PwmDma::get_compare (
            float duty_cycle ) const
```

Get the compare value for ad duty cycle.

Parameters

duty_cycle	Duty cycle to get the compare value for

Returns

uint32_t Compare value for the duty cycle

7.55.3.2 start_dma()

```
void hal::PwmDma::start_dma (
            uint32_t buffer[],
            uint32_t size )
```

Start PWM and DMA transfer.

Parameters

buffer	Buffer to transfer
size	Size of the buffer

7.55.3.3 stop_dma()

```
void hal::PwmDma::stop_dma ( )
```

Stop PWM and DMA transfer.

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

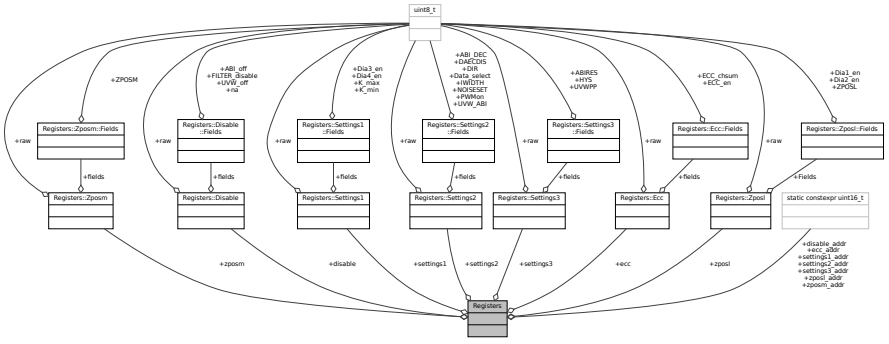
- inc/hal/pwm_dma.hpp
- src/hal/pwm_dma.cpp

7.56 Registers Struct Reference

Registers class for the rotary sensor configuration.

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers:



Classes

- union Disable

  Registers union types definition.
- union Ecc
- union Settings1
- union Settings2
- union Settings3
- union Zposl
- union Zposm

Public Attributes

- Disable disable

  Member variables to be configured in the rotary sensor.
- Zposm zposm
- Zposl zposl
- Settings1 settings1
- Settings2 settings2
- Settings3 settings3
- Ecc ecc

Static Public Attributes

- static constexpr uint16_t disable_addr {0x0015}

  Register addresses in the rotary sensor memory.
- static constexpr uint16_t zposm_addr {0x0016}
- static constexpr uint16_t zposl_addr {0x0017}
- static constexpr uint16_t settings1_addr {0x0018}
- static constexpr uint16_t settings2_addr {0x0019}
- static constexpr uint16_t settings3_addr {0x001A}
- static constexpr uint16_t ecc_addr {0x001B}

7.56.1 Detailed Description

Registers class for the rotary sensor configuration.

7.56.2 Member Data Documentation

7.56.2.1 disable

```
Disable Registers::disable
```

Member variables to be configured in the rotary sensor.

7.56.2.2 disable_addr

```
constexpr uint16_t Registers::disable_addr {0x0015}  [static], [constexpr]
```

Register addresses in the rotary sensor memory.

7.56.2.3 ecc

```
Ecc Registers::ecc
```

7.56.2.4 ecc_addr

```
constexpr uint16_t Registers::ecc_addr {0x001B}  [static], [constexpr]
```

7.56.2.5 settings1

```
Settings1 Registers::settings1
```

7.56.2.6 settings1_addr

```
constexpr uint16_t Registers::settings1_addr {0x0018}  [static], [constexpr]
```

7.56.2.7 settings2

```
Settings2 Registers::settings2
```

7.56.2.8 settings2_addr

```
constexpr uint16_t Registers::settings2_addr {0x0019}  [static], [constexpr]
```

7.56.2.9 settings3

```
Settings3 Registers::settings3
```

7.56.2.10 settings3_addr

```
constexpr uint16_t Registers::settings3_addr {0x001A}  [static], [constexpr]
```

7.56.2.11 zposl

```
Zposl Registers::zposl
```

7.56.2.12 zposl_addr

```
constexpr uint16_t Registers::zposl_addr {0x0017}  [static], [constexpr]
```

7.56.2.13 zposm

```
Zposm Registers::zposm
```

7.56.2.14 zposm_addr

```
constexpr uint16_t Registers::zposm_addr {0x0016}  [static], [constexpr]
```

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

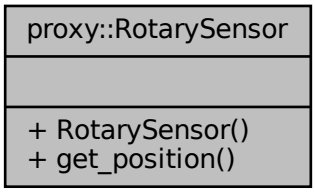
- inc/proxy/rotary_sensor_reg.hpp

7.57 proxy::RotarySensor Class Reference

Class to handle rotary sensor peripheral on STM32 microcontrollers.

`#include <rotary_sensor.hpp>`

Collaboration diagram for proxy::RotarySensor:



Classes

- struct Config

  Rotary sensor configuration struct.

Public Member Functions

- RotarySensor (const Config &config)

  Construct a new RotarySensor object.
- float get_position () const

  Get the rotary sensor position over an axis.

7.57.1 Detailed Description

Class to handle rotary sensor peripheral on STM32 microcontrollers.

7.57.2 Constructor & Destructor Documentation

7.57.2.1 RotarySensor()

```
proxy::RotarySensor::RotarySensor (
            const Config & config )  [explicit]
```

Construct a new RotarySensor object.

Parameters

config	Configuration for the rotary sensor

7.57.3 Member Function Documentation

7.57.3.1 get_position()

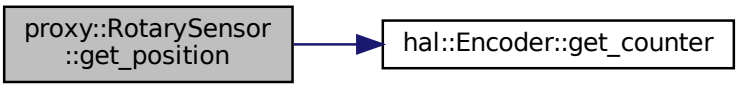
```
float proxy::RotarySensor::get_position ( ) const
```

Get the rotary sensor position over an axis.

Returns

float Current angular position of the sensor in radians

Here is the call graph for this function:



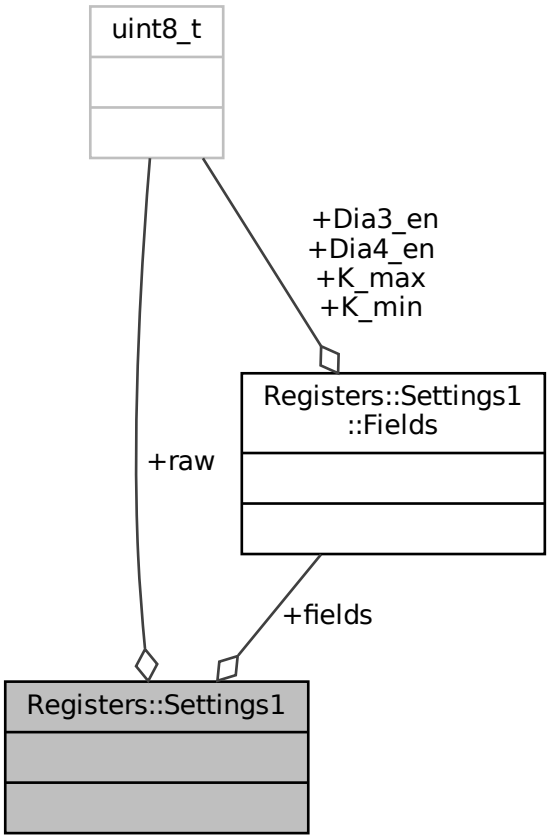
The documentation for this class was generated from the following files:

- inc/proxy/rotary_sensor.hpp
- src/proxy/rotary_sensor.cpp

7.58 Registers::Settings1 Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings1:



Classes

- struct Fields

Public Attributes

- Fields fields
- uint8_t raw

7.58.1 Member Data Documentation

7.58.1.1 fields

```
Fields Registers::Settings1::fields
```

7.58.1.2 raw

```
uint8_t Registers::Settings1::raw
```

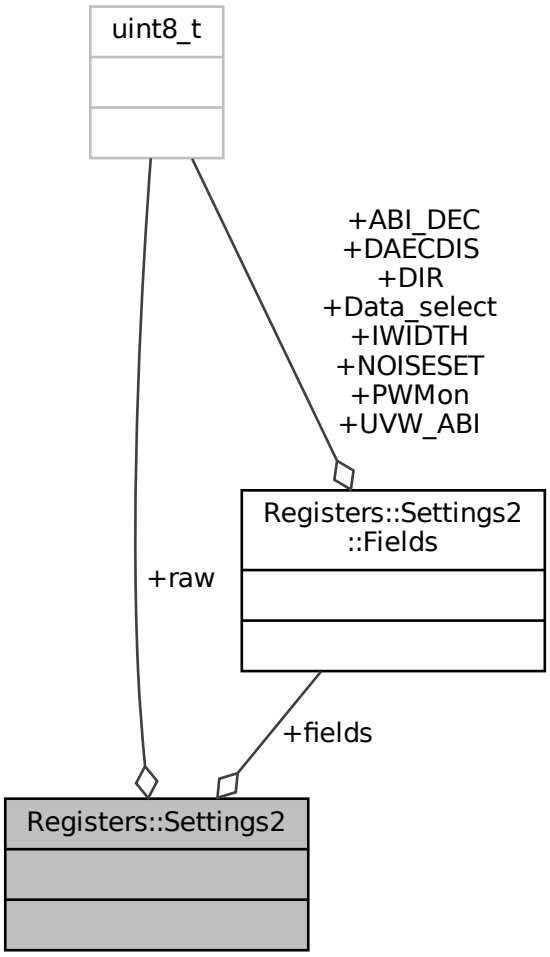
The documentation for this union was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.59 Registers::Settings2 Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings2:



Classes

- struct Fields

Public Attributes

- Fields fields
- uint8_t raw

7.59.1 Member Data Documentation

7.59.1.1 fields

```
Fields Registers::Settings2::fields
```

7.59.1.2 raw

```
uint8_t Registers::Settings2::raw
```

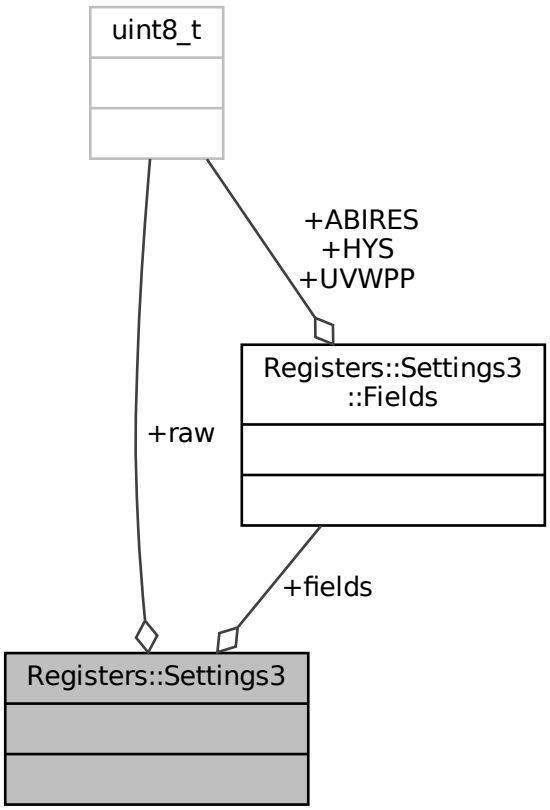
The documentation for this union was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.60 Registers::Settings3 Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Settings3:



Classes

- struct Fields

Public Attributes

- Fields fields
- uint8_t raw

7.60.1 Member Data Documentation

7.60.1.1 fields

```
Fields Registers::Settings3::fields
```

7.60.1.2 raw

```
uint8_t Registers::Settings3::raw
```

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

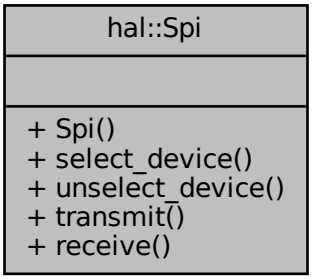
- inc/proxy/rotary_sensor_reg.hpp

7.61 hal::Spi Class Reference

Class to handle SPI peripheral on STM32 microcontrollers.

```
#include <spi.hpp>
```

Collaboration diagram for hal::Spi:



Classes

- struct Config

  SPI configuration struct.

Public Member Functions

- Spi (const Config &config)

  Construct a new Spi object.
- bool select_device ()

  Activate the chip select.
- void unselect_device ()

  Deactivate the chip select.
- void transmit (uint8_t data[ ], uint32_t size)

  Transmit data over SPI.
- void receive (uint8_t data[ ], uint32_t size)

  Receive data over SPI.

7.61.1 Detailed Description

Class to handle SPI peripheral on STM32 microcontrollers.

7.61.2 Constructor & Destructor Documentation

7.61.2.1 Spi()

```
hal::Spi::Spi (
            const Config & config )  [explicit]
```

Construct a new Spi object.

Parameters

config	Configuration for the SPI

7.61.3 Member Function Documentation

7.61.3.1 receive()

```
void hal::Spi::receive (
            uint8_t data[],
            uint32_t size )
```

Receive data over SPI.

Parameters

data	Data to receive data
size	Size of the data

7.61.3.2 select_device()

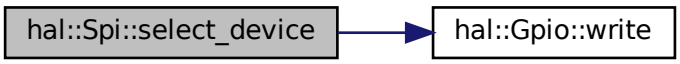
```
bool hal::Spi::select_device ( )
```

Activate the chip select.

Returns

bool True if the device was successfully selected, false otherwise

Here is the call graph for this function:



7.61.3.3 transmit()

```
void hal::Spi::transmit (
            uint8_t data[],
            uint32_t size )
```

Transmit data over SPI.

Parameters

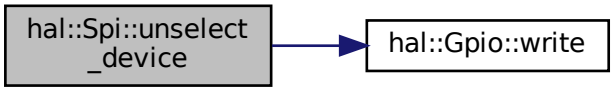
data	Data to transmit
size	Size of the buffer

7.61.3.4 unselect_device()

```
void hal::Spi::unselect_device ( )
```

Deactivate the chip select.

Here is the call graph for this function:



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

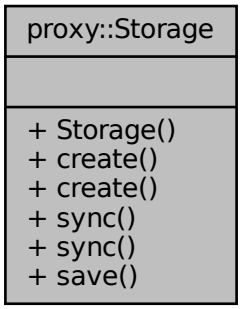
- inc/hal/spi.hpp
- src/hal/spi.cpp

7.62 proxy::Storage Class Reference

Class for controlling the storage.

`#include <storage.hpp>`

Collaboration diagram for proxy::Storage:



Classes

- struct Config

  Configuration structure for the storage.

Public Member Functions

- Storage (const Config &config)

  Constructor for the Storage class.
- template<Fundamental T>
  void create (const std::string &name, const T &data)

  Create a new primitive variable in the storage.
- void create (const std::string &name, const ISerializable &data)

  Create a new serializable variable in the storage.
- template<Fundamental T>
  void sync (const std::string &name, T &data)

  Sync a primitive variable with the storage.
- void sync (const std::string &name, ISerializable &data)

  Sync a serializable variable with the storage.
- void save ()

  Save the storage to the flash.

7.62.1 Detailed Description

Class for controlling the storage.

7.62.2 Constructor & Destructor Documentation

7.62.2.1 Storage()

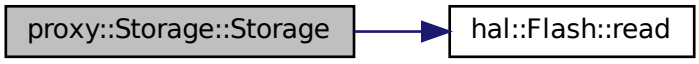
```
proxy::Storage::Storage (
            const Config & config )  [explicit]
```

Constructor for the Storage class.

Parameters

config	Configuration for the storage

Here is the call graph for this function:



7.62.3 Member Function Documentation

7.62.3.1 create() [1/2]

```
void proxy::Storage::create (
            const std::string & name,
            const ISerializable & data )
```

Create a new serializable variable in the storage.

Parameters

name	Name of the variable
data	Reference to the variable

7.62.3.2 create() [2/2]

```
template<Fundamental T>
void proxy::Storage::create (
            const std::string & name,
            const T & data )
```

Create a new primitive variable in the storage.

Template Parameters

T	Type of the variable

Parameters

name	Name of the variable
data	Reference to the variable

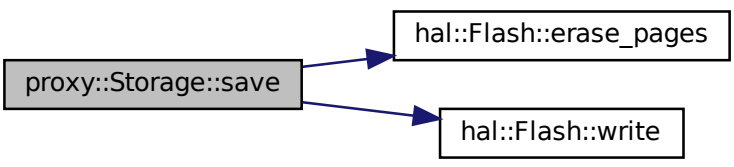
Here is the caller graph for this function:

7.62.3.3 save()

```
void proxy::Storage::save ( )
```

Save the storage to the flash.

Here is the call graph for this function:



7.62.3.4 sync() [1/2]

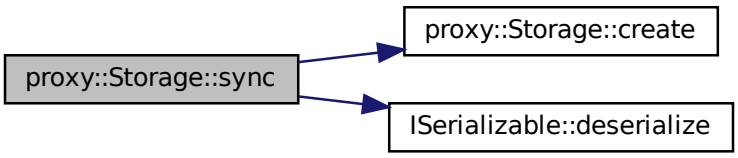
```
void proxy::Storage::sync (
            const std::string & name,
            ISerializable & data )
```

Sync a serializable variable with the storage.

Parameters

name	Name of the variable
data	Reference to the variable

Here is the call graph for this function:



7.62.3.5 sync() [2/2]

```
template<Fundamental T>
void proxy::Storage::sync (
            const std::string & name,
            T & data )
```

Sync a primitive variable with the storage.

Template Parameters

T	Type of the variable

Parameters

name	Name of the variable
data	Reference to the variable

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

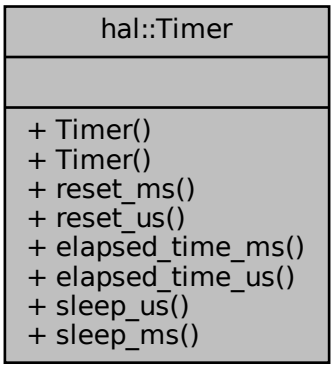
- inc/proxy/storage.hpp
- src/proxy/storage.cpp

7.63 hal::Timer Class Reference

Class to handle timer peripheral on STM32 microcontrollers.

`#include <timer.hpp>`

Collaboration diagram for hal::Timer:



Classes

- struct Config

  Timer configuration struct.

Public Member Functions

- Timer ()=default

  Construct a new Timer object.
- Timer (const Config &config)

  Construct a new Timer object.
- void reset_ms ()

  Reset the timer counter in milliseconds.
- void reset_us ()

  Reset the timer counter in microseconds.
- uint32_t elapsed_time_ms () const

  Get the time elapsed since the last reset.
- uint32_t elapsed_time_us () const

  Get the time elapsed since the last reset.
- void sleep_us (uint32_t time) const

  Sleep for a given amount of time.

Static Public Member Functions

- static void sleep_ms (uint32_t time)

  Sleep for a given amount of time.

7.63.1 Detailed Description

Class to handle timer peripheral on STM32 microcontrollers.

7.63.2 Constructor & Destructor Documentation

7.63.2.1 Timer() [1/2]

```
hal::Timer::Timer ( )  [default]
```

Construct a new Timer object.

7.63.2.2 Timer() [2/2]

```
hal::Timer::Timer (
            const Config & config )  [explicit]
```

Construct a new Timer object.

Parameters

config	Configuration for the timer

7.63.3 Member Function Documentation

7.63.3.1 elapsed_time_ms()

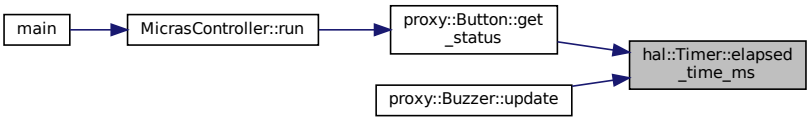
```
uint32_t hal::Timer::elapsed_time_ms ( ) const
```

Get the time elapsed since the last reset.

Returns

uint32_t Time elapsed in miliseconds

Here is the caller graph for this function:



7.63.3.2 elapsed_time_us()

```
uint32_t hal::Timer::elapsed_time_us ( ) const
```

Get the time elapsed since the last reset.

Returns

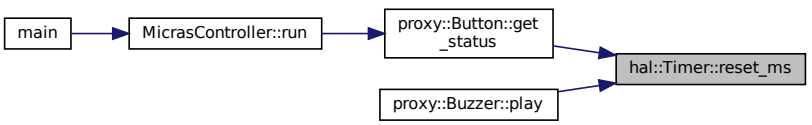
uint32_t Time elapsed in microseconds

7.63.3.3 reset_ms()

```
void hal::Timer::reset_ms ( )
```

Reset the timer counter in milliseconds.

Here is the caller graph for this function:



7.63.3.4 reset_us()

```
void hal::Timer::reset_us ( )
```

Reset the timer counter in microseconds.

7.63.3.5 sleep_ms()

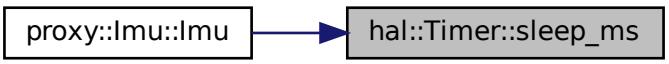
```
void hal::Timer::sleep_ms (
            uint32_t time )  [static]
```

Sleep for a given amount of time.

Parameters

time	Time to sleep in milliseconds

Here is the caller graph for this function:



7.63.3.6 sleep_us()

```
void hal::Timer::sleep_us (
            uint32_t time ) const
```

Sleep for a given amount of time.

Parameters

time	Time to sleep in microseconds

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

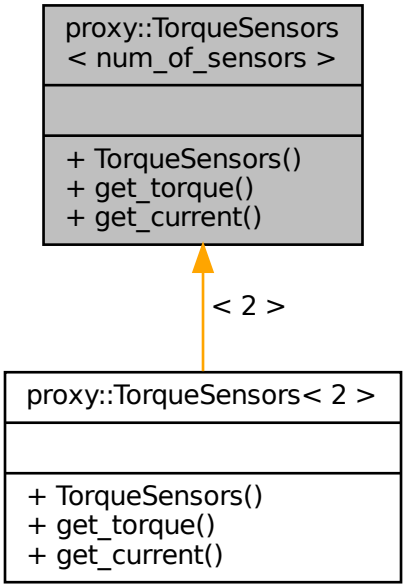
- inc/hal/timer.hpp
- src/hal/timer.cpp

7.64 proxy::TorqueSensors< num_of_sensors > Class Template Reference

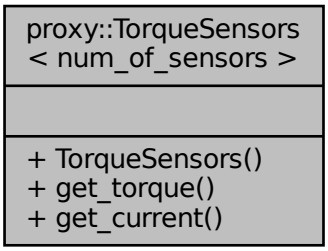
Class for controlling TorqueSensors.

`#include <torque_sensors.hpp>`

Inheritance diagram for proxy::TorqueSensors< num_of_sensors >:



Collaboration diagram for proxy::TorqueSensors< num_of_sensors >:



Classes

- struct Config

  Configuration structure for torque sensors.

Public Member Functions

- TorqueSensors (const Config &config)

  Constructor for the TorqueSensors class.
- float get_torque (uint8_t sensor_index) const

  Get the torque from the sensor.
- float get_current (uint8_t sensor_index) const

  Get the current from the sensor.

7.64.1 Detailed Description

template<uint8_t num_of_sensors>
class proxy::TorqueSensors< num_of_sensors >

Class for controlling TorqueSensors.

7.64.2 Constructor & Destructor Documentation

7.64.2.1 TorqueSensors()

```
template<uint8_t num_of_sensors>
proxy::TorqueSensors< num_of_sensors >::TorqueSensors (
            const Config & config )  [explicit]
```

Constructor for the TorqueSensors class.

Parameters

config	Configuration for the torque sensors

7.64.3 Member Function Documentation

7.64.3.1 get_current()

```
template<uint8_t num_of_sensors>
float proxy::TorqueSensors< num_of_sensors >::get_current (
            uint8_t sensor_index ) const
```

Get the current from the sensor.

Parameters

sensor_index	Index of the sensor

Returns

float Current reading from the sensor in amps

7.64.3.2 get_torque()

```
template<uint8_t num_of_sensors>
float proxy::TorqueSensors< num_of_sensors >::get_torque (
            uint8_t sensor_index ) const
```

Get the torque from the sensor.

Parameters

sensor_index	Index of the sensor

Returns

float Torque reading from the sensor in N∗m

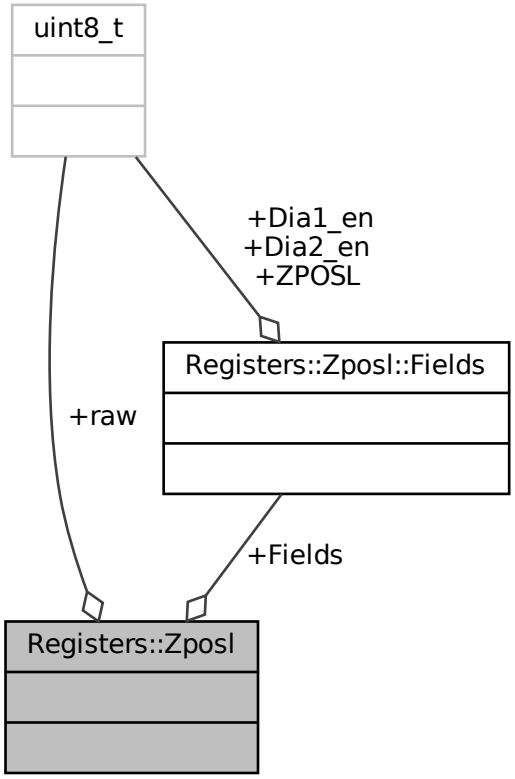
The documentation for this class was generated from the following files:

- inc/proxy/torque_sensors.hpp
- src/proxy/torque_sensors.cpp

7.65 Registers::Zposl Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposl:



Classes

- struct Fields

Public Attributes

- Fields Fields
- uint8_t raw

7.65.1 Member Data Documentation

7.65.1.1 Fields

```
Fields Registers::Zposl::Fields
```

7.65.1.2 raw

```
uint8_t Registers::Zposl::raw
```

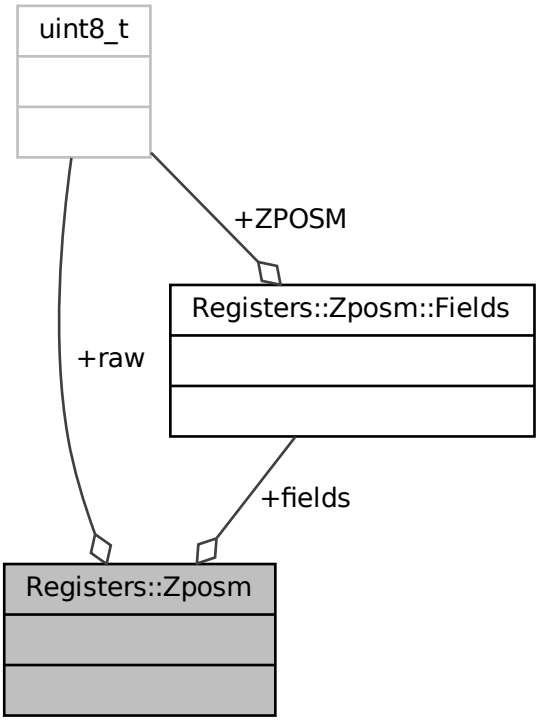
The documentation for this union was generated from the following file:

- inc/proxy/rotary_sensor_reg.hpp

7.66 Registers::Zposm Union Reference

```
#include <rotary_sensor_reg.hpp>
```

Collaboration diagram for Registers::Zposm:



Classes

- struct Fields

Public Attributes

- Fields fields
- uint8_t raw

7.66.1 Member Data Documentation

7.66.1.1 fields

```
Fields Registers::Zposm::fields
```

7.66.1.2 raw

```
uint8_t Registers::Zposm::raw
```

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

- inc/proxy/rotary_sensor_reg.hpp

Chapter 8

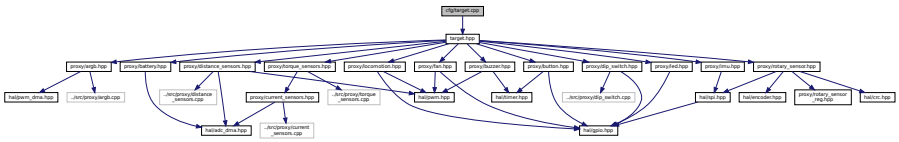
File Documentation

8.1 cfg/target.cpp File Reference

Target specific configuration.

```
#include "target.hpp"
```

Include dependency graph for target.cpp:



Variables

- const proxy::Argb< 2 >::Config argb_config
- const proxy::Battery::Config battery_config
- const proxy::Button::Config button_config
- const proxy::Buzzer::Config buzzer_config
- const proxy::DipSwitch< 4 >::Config dip_switch_config
- const proxy::DistanceSensors< 4 >::Config distance_sensors_config
- const proxy::Fan::Config fan_config
- const proxy::Imu::Config imu_config
- const proxy::Led::Config led_config
- const proxy::Locomotion::Config locomotion_config
- const proxy::RotarySensor::Registers rotary_sensor_reg_config
- const proxy::RotarySensor::Config rotary_sensor_left_config
- const proxy::RotarySensor::Config rotary_sensor_right_config
- const proxy::TorqueSensors< 2 >::Config torque_sensors_config

8.1.1 Detailed Description

Target specific configuration.

Date

03/2024

8.1.2 Variable Documentation

8.1.2.1 argb_config

```
const proxy::Argb<2>::Config argb_config
```

Initial value:

```
= {
    .pwm = {
        .handle = &htim8,
        .init_function = MX_TIM8_Init,
        .timer_channel = TIM_CHANNEL_1
    }
}
```

8.1.2.2 battery_config

```
const proxy::Battery::Config battery_config
```

Initial value:

```
= {
    .adc = {
        .handle = &hadc3,
        .init_function = MX_ADC3_Init,
        .max_reading = 4095,
        .reference_voltage = 3.0F
    },
    .voltage_divider = 3.0F
}
```

8.1.2.3 button_config

```
const proxy::Button::Config button_config
```

Initial value:

```
= {
    .gpio = {
        .port = GPIOA,
        .pin = GPIO_PIN_12
    },
    .pull_resistor = proxy::Button::PullResistor::PULL_UP
}
```

8.1.2.4 buzzer_config

```
const proxy::Buzzer::Config buzzer_config
```

Initial value:

```
= {
    .pwm = {
        .handle = &htim4,
        .init_function = MX_TIM4_Init,
        .timer_channel = TIM_CHANNEL_1
    }
}
```

8.1.2.5 dip_switch_config

```
const proxy::DipSwitch<4>::Config dip_switch_config
```

Initial value:

```
= {
    .gpio_array = {{
        {
            .port = GPIOC,
            .pin = GPIO_PIN_7
        },
        {
            .port = GPIOB,
            .pin = GPIO_PIN_15
        },
        {
            .port = GPIOB,
            .pin = GPIO_PIN_14
        },
        {
            .port = GPIOB,
            .pin = GPIO_PIN_13
        }
    }}
}
```

8.1.2.6 distance_sensors_config

```
const proxy::DistanceSensors<4>::Config distance_sensors_config
```

Initial value:

```
= {
    .adc = {
        .handle = &hadc1,
        .init_function = MX_ADC1_Init,
        .max_reading = 4095,
        .reference_voltage = 3.3F
    },
    .led_pwm = {
        .handle = &htim15,
        .init_function = MX_TIM15_Init,
        .timer_channel = TIM_CHANNEL_1
    },
    .max_distance = 0.3F
}
```

8.1.2.7 fan_config

```
const proxy::Fan::Config fan_config
```

Initial value:

```
= {
    .pwm = {
        .handle = &htim17,
        .init_function = MX_TIM17_Init,
        .timer_channel = TIM_CHANNEL_1
    },
    .direction_gpio = {
        .port = GPIOC,
        .pin = GPIO_PIN_13
    },
    .enable_gpio = {
        .port = GPIOB,
        .pin = GPIO_PIN_7
    }
}
```

8.1.2.8 imu_config

```
const proxy::Imu::Config imu_config
```

Initial value:

```
= {
    .spi = {
        .handle = &hspi1,
        .init_function = MX_SPI1_Init,
        .gpio = {
            .port = GPIOB,
            .pin = GPIO_PIN_6
        },
        .timeout = 2
    },
    .gyroscope_data_rate = LSM6DSV_ODR_AT_480Hz,
    .accelerometer_data_rate = LSM6DSV_ODR_AT_480Hz,
    .orientation_data_rate = LSM6DSV_SFLP_480Hz,
    .gyroscope_scale = LSM6DSV_4000dps,
    .accelerometer_scale = LSM6DSV_8g,
    .gyroscope_filter = LSM6DSV_GY_ULTRA_LIGHT,
    .accelerometer_filter = LSM6DSV_XL_ULTRA_LIGHT
}
```

8.1.2.9 led_config

```
const proxy::Led::Config led_config
```

Initial value:

```
= {
    .gpio = {
        .port = GPIOA,
        .pin = GPIO_PIN_15
    }
}
```

8.1.2.10 locomotion_config

```
const proxy::Locomotion::Config locomotion_config
```

Initial value:

```
= {
    .pwm_left_fwd = {
        .handle = &htim4,
        .init_function = MX_TIM4_Init,
        .timer_channel = TIM_CHANNEL_4
    },
    .pwm_left_bwd = {
        .handle = &htim4,
        .init_function = MX_TIM4_Init,
        .timer_channel = TIM_CHANNEL_3
    },
    .pwm_right_fwd = {
        .handle = &htim1,
        .init_function = MX_TIM1_Init,
        .timer_channel = TIM_CHANNEL_2
    },
    .pwm_right_bwd = {
        .handle = &htim1,
        .init_function = MX_TIM1_Init,
        .timer_channel = TIM_CHANNEL_1
    },
    .enable_gpio = {
        .port = GPIOA,
        .pin = GPIO_PIN_10
    }
}
```

8.1.2.11 rotary_sensor_left_config

```
const proxy::RotarySensor::Config rotary_sensor_left_config
```

Initial value:

```
= {
    .spi = {
        .handle = &hspi1,
        .init_function = MX_SPI1_Init,
        .gpio = {
            .port = GPIOA,
            .pin = GPIO_PIN_6
        },
        .timeout = 2
    },
    .encoder = {
        .handle = &htim2,
        .init_function = MX_TIM2_Init,
        .timer_channel = TIM_CHANNEL_ALL
    },
    .crc = {
        .handle = &hcrc
    },
    .resolution = 4096,
    .registers = rotary_sensor_reg_config
}
```

8.1.2.12 rotary_sensor_reg_config

```
const proxy::RotarySensor::Registers rotary_sensor_reg_config
```

8.1.2.13 rotary_sensor_right_config

```
const proxy::RotarySensor::Config rotary_sensor_right_config
```

Initial value:

```
= {
    .spi = {
        .handle = &hspi1,
        .init_function = MX_SPI1_Init,
        .gpio = {
            .port = GPIOB,
            .pin = GPIO_PIN_1
        },
        .timeout = 2
    },
    .encoder = {
        .handle = &htim5,
        .init_function = MX_TIM5_Init,
        .timer_channel = TIM_CHANNEL_ALL
    },
    .crc = {
        .handle = &hcrc
    },
    .resolution = 4096,
    .registers = rotary_sensor_reg_config
}
```

8.1.2.14 torque_sensors_config

```
const proxy::TorqueSensors<2>::Config torque_sensors_config
```

Initial value:

```
= {
    .current_sensors = {
        .adc = {
            .handle = &hadc2,
            .init_function = MX_ADC2_Init,
            .max_reading = 4095,
            .reference_voltage = 3.3F
        },
        .shunt_resistor = 0.04F
    },
    .max_torque = 0.5F
}
```

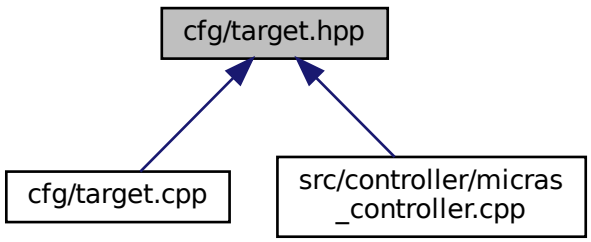
8.2 cfg/target.hpp File Reference

Target specific configuration.

```
#include "proxy/argb.hpp"
#include "proxy/battery.hpp"
#include "proxy/button.hpp"
#include "proxy/buzzer.hpp"
#include "proxy/dip_switch.hpp"
#include "proxy/distance_sensors.hpp"
#include "proxy/fan.hpp"
#include "proxy/imu.hpp"
#include "proxy/led.hpp"
#include "proxy/locomotion.hpp"
#include "proxy/rotary_sensor.hpp"
#include "proxy/torque_sensors.hpp"
```

Include dependency graph for target.hpp:

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



Variables

- const proxy::Argb< 2 >::Config argb_config
- const proxy::Battery::Config battery_config
- const proxy::Button::Config button_config
- const proxy::Buzzer::Config buzzer_config
- const proxy::DipSwitch< 4 >::Config dip_switch_config
- const proxy::DistanceSensors< 4 >::Config distance_sensors_config
- const proxy::Fan::Config fan_config
- const proxy::Imu::Config imu_config
- const proxy::Led::Config led_config
- const proxy::Locomotion::Config locomotion_config
- const proxy::RotarySensor::Config rotary_sensor_left_config
- const proxy::RotarySensor::Config rotary_sensor_right_config
- const proxy::TorqueSensors< 2 >::Config torque_sensors_config

8.2.1 Detailed Description

Target specific configuration.

Date

03/2024

8.2.2 Variable Documentation

8.2.2.1 argb_config

```
const proxy::Argb<2>::Config argb_config  [extern]
```

8.2.2.2 battery_config

```
const proxy::Battery::Config battery_config  [extern]
```

8.2.2.3 button_config

```
const proxy::Button::Config button_config  [extern]
```

8.2.2.4 buzzer_config

```
const proxy::Buzzer::Config buzzer_config  [extern]
```

8.2.2.5 dip_switch_config

```
const proxy::DipSwitch<4>::Config dip_switch_config  [extern]
```

8.2.2.6 distance_sensors_config

```
const proxy::DistanceSensors<4>::Config distance_sensors_config  [extern]
```

8.2.2.7 fan_config

```
const proxy::Fan::Config fan_config  [extern]
```

8.2.2.8 imu_config

```
const proxy::Imu::Config imu_config  [extern]
```

8.2.2.9 led_config

```
const proxy::Led::Config led_config  [extern]
```

8.2.2.10 locomotion_config

```
const proxy::Locomotion::Config locomotion_config  [extern]
```

8.2.2.11 rotary_sensor_left_config

```
const proxy::RotarySensor::Config rotary_sensor_left_config  [extern]
```

8.2.2.12 rotary_sensor_right_config

```
const proxy::RotarySensor::Config rotary_sensor_right_config  [extern]
```

8.2.2.13 torque_sensors_config

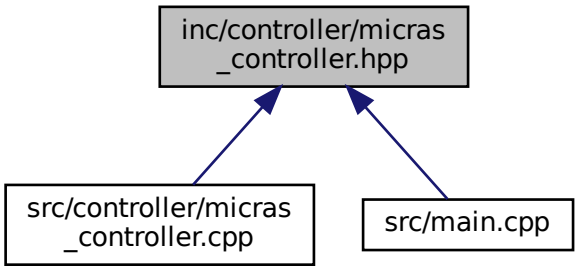
```
const proxy::TorqueSensors<2>::Config torque_sensors_config  [extern]
```

8.3 inc/controller/micras_controller.hpp File Reference

```
#include "proxy/argb.hpp"
#include "proxy/battery.hpp"
#include "proxy/button.hpp"
#include "proxy/buzzer.hpp"
#include "proxy/dip_switch.hpp"
#include "proxy/distance_sensors.hpp"
#include "proxy/fan.hpp"
#include "proxy/imu.hpp"
#include "proxy/led.hpp"
#include "proxy/locomotion.hpp"
#include "proxy/rotary_sensor.hpp"
#include "proxy/torque_sensors.hpp"
```

Include dependency graph for micras_controller.hpp:

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



Classes

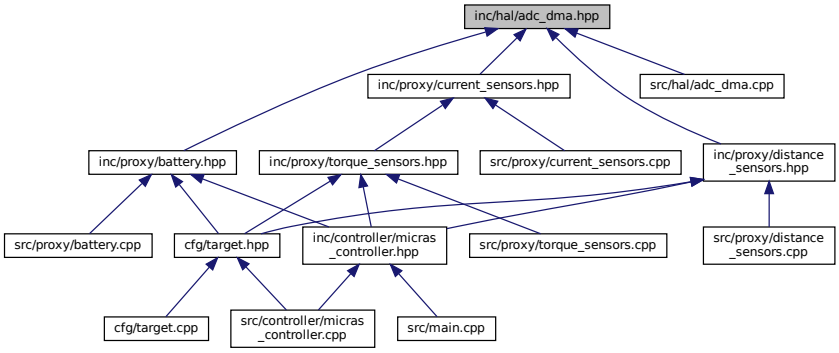
- class MicrasController

  Class for controlling the Micras robot.

8.4 inc/hal/adc_dma.hpp File Reference

ADC DMA HAL header.

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



Classes

- class hal::AdcDma

  Class to handle ADC peripheral on STM32 microcontrollers using DMA.
- struct hal::AdcDma::Config

  Configuration structure for ADC DMA.

Namespaces

- hal

8.4.1 Detailed Description

ADC DMA HAL header.

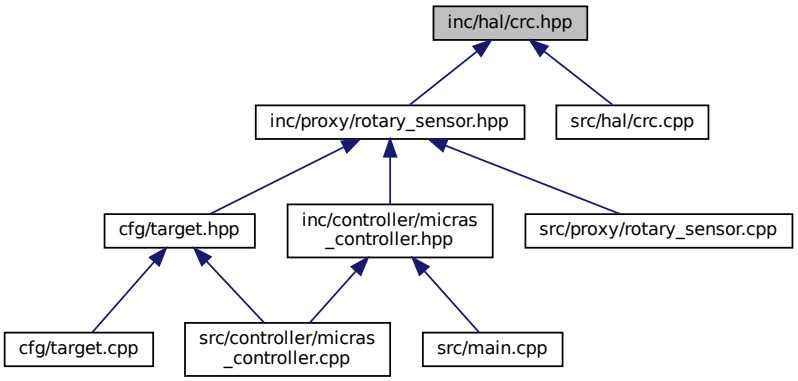
Date

03/2024

8.5 inc/hal/crc.hpp File Reference

STM32 CRC HAL wrapper.

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



Classes

- class hal::Crc

  Class to handle the cyclic redundancy check peripheral on STM32 microcontrollers.
- struct hal::Crc::Config

  CRC configuration struct.

Namespaces

- hal

8.5.1 Detailed Description

STM32 CRC HAL wrapper.

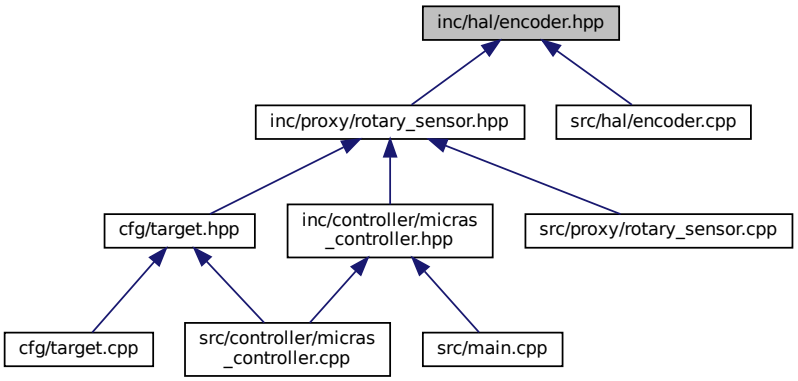
Date

03/2024

8.6 inc/hal/encoder.hpp File Reference

STM32 encoder HAL wrapper.

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



Classes

- class hal::Encoder

  Class to handle encoder peripheral on STM32 microcontrollers.
- struct hal::Encoder::Config

  Encoder configuration struct.

Namespaces

- hal

8.6.1 Detailed Description

STM32 encoder HAL wrapper.

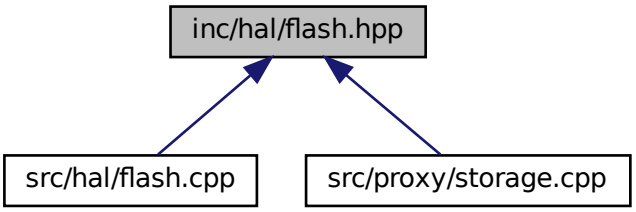
Date

03/2024

8.7 inc/hal/flash.hpp File Reference

STM32 flash HAL wrapper.

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



Classes

- class hal::Flash

  Class to handle flash memory on STM32 microcontrollers.

Namespaces

- hal

8.7.1 Detailed Description

STM32 flash HAL wrapper.

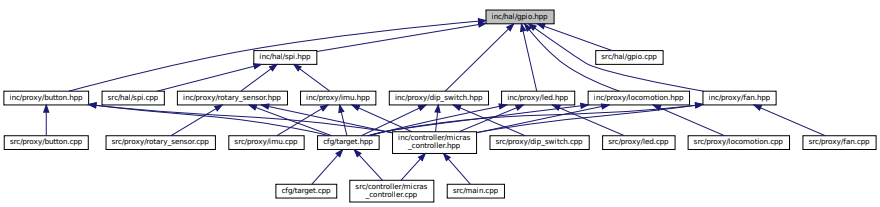
Date

03/2024

8.8 inc/hal/gpio.hpp File Reference

HAL GPIO class header.

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



Classes

- class hal::Gpio

  Class for controlling GPIO pins on STM32 microcontrollers.
- struct hal::Gpio::Config

  Configuration structure for GPIO pin.

Namespaces

- hal

8.8.1 Detailed Description

HAL GPIO class header.

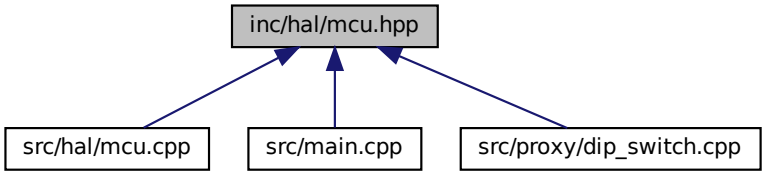
Date

03/2024

8.9 inc/hal/mcu.hpp File Reference

MCU related.

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



Classes

- class hal::Mcu

  Microcontroller unit class.

Namespaces

- hal

8.9.1 Detailed Description

MCU related.

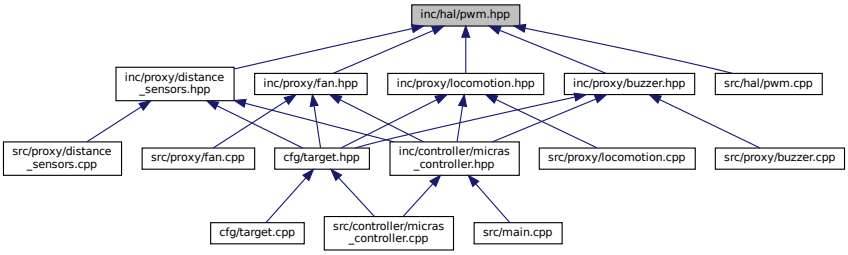
Date

03/2024

8.10 inc/hal/pwm.hpp File Reference

STM32 PWM HAL wrapper.

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



Classes

- class hal::Pwm

  Class to handle PWM peripheral on STM32 microcontrollers.
- struct hal::Pwm::Config

  PWM configuration struct.

Namespaces

- hal

8.10.1 Detailed Description

STM32 PWM HAL wrapper.

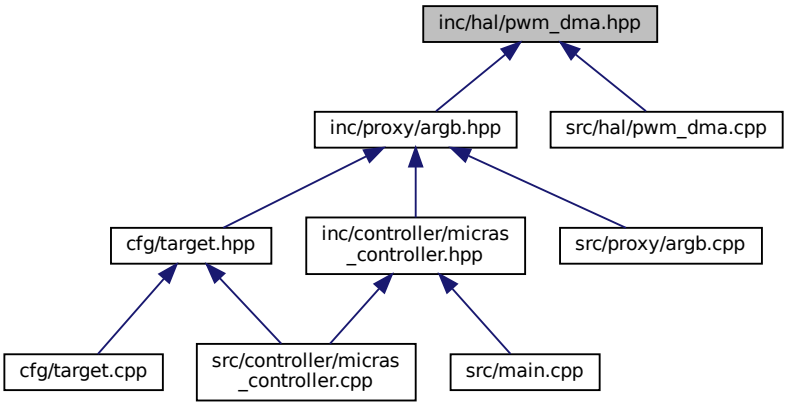
Date

03/2024

8.11 inc/hal/pwm_dma.hpp File Reference

STM32 PWM DMA HAL wrapper.

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



Classes

- class hal::PwmDma

  Class to handle PWM peripheral on STM32 microcontrollers using DMA.
- struct hal::PwmDma::Config

  PWM configuration struct.

Namespaces

- hal

8.11.1 Detailed Description

STM32 PWM DMA HAL wrapper.

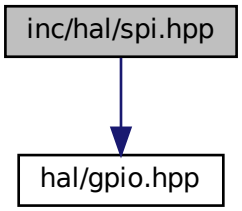
Date

03/2024

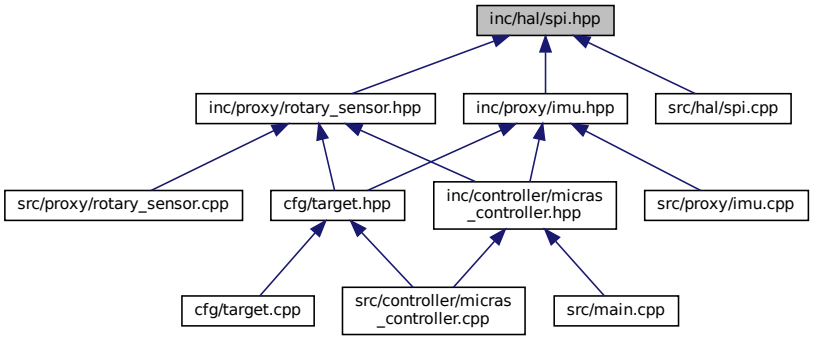
8.12 inc/hal/spi.hpp File Reference

STM32 SPI HAL wrapper.

`#include "hal/gpio.hpp"`
Include dependency graph for spi.hpp:



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



Classes

- class hal::Spi

  Class to handle SPI peripheral on STM32 microcontrollers.
- struct hal::Spi::Config

  SPI configuration struct.

Namespaces

- hal

8.12.1 Detailed Description

STM32 SPI HAL wrapper.

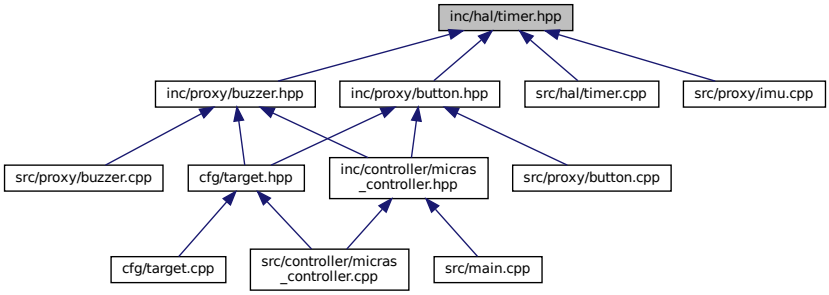
Date

03/2024

8.13 inc/hal/timer.hpp File Reference

STM32 Timer HAL wrapper.

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



Classes

- class hal::Timer

  Class to handle timer peripheral on STM32 microcontrollers.
- struct hal::Timer::Config

  Timer configuration struct.

Namespaces

- hal

8.13.1 Detailed Description

STM32 Timer HAL wrapper.

Date

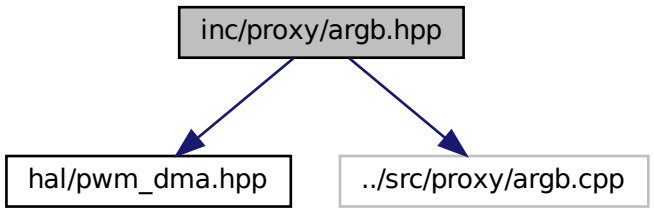
03/2024

8.14 inc/proxy/argb.hpp File Reference

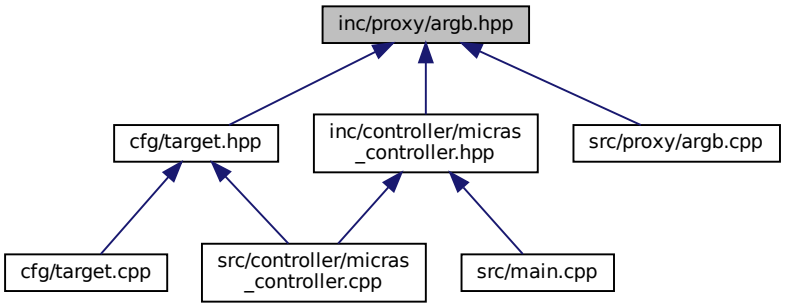
Proxy Argb class declaration.

```
#include "hal/pwm_dma.hpp"
#include "../src/proxy/argb.cpp"
```

Include dependency graph for argb.hpp:



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



Classes

- class proxy::Argb< num_of_leds >

  Class for controlling an addressable RGB LED.
- struct proxy::Argb< num_of_leds >::Config

  Configuration structure for the addressable RGB LED.
- struct proxy::Argb< num_of_leds >::Color

  Structure for storing color information.

Namespaces

- proxy

8.14.1 Detailed Description

Proxy Argb class declaration.

Date

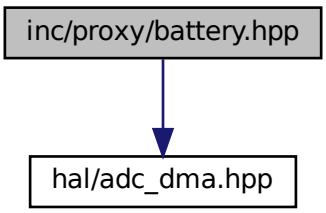
03/2024

8.15 inc/proxy/battery.hpp File Reference

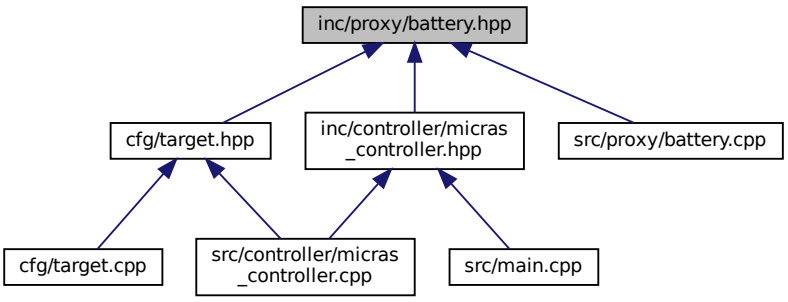
Proxy Battery class declaration.

```
#include "hal/adc_dma.hpp"
```

Include dependency graph for battery.hpp:



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



Classes

- class proxy::Battery

  Class for getting the battery voltage.
- struct proxy::Battery::Config

  Configuration structure for the battery.

Namespaces

- proxy

8.15.1 Detailed Description

Proxy Battery class declaration.

Date

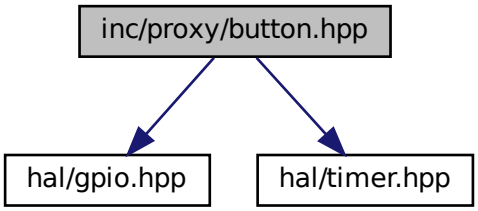
03/2024

8.16 inc/proxy/button.hpp File Reference

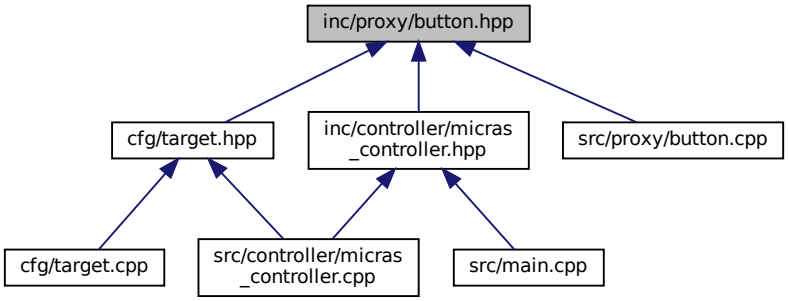
Proxy Button class header.

```
#include "hal/gpio.hpp"
#include "hal/timer.hpp"
```

Include dependency graph for button.hpp:



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



Classes

- class proxy::Button

  Class for controlling a button.
- struct proxy::Button::Config

  Configuration structure for button.

Namespaces

- proxy

8.16.1 Detailed Description

Proxy Button class header.

Date

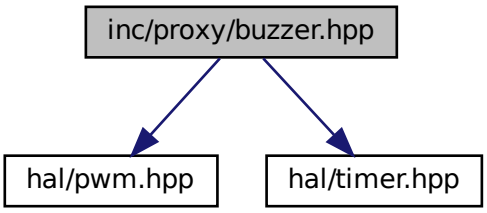
03/2024

8.17 inc/proxy/buzzer.hpp File Reference

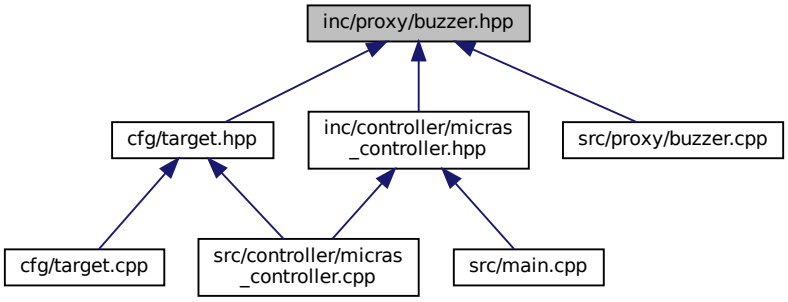
Proxy Buzzer class declaration.

```
#include "hal/pwm.hpp"
#include "hal/timer.hpp"
```

Include dependency graph for buzzer.hpp:



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



Classes

- class proxy::Buzzer

  Class for controlling a buzzer.
- struct proxy::Buzzer::Config

  Configuration structure for the buzzer.

Namespaces

- proxy

8.17.1 Detailed Description

Proxy Buzzer class declaration.

Date

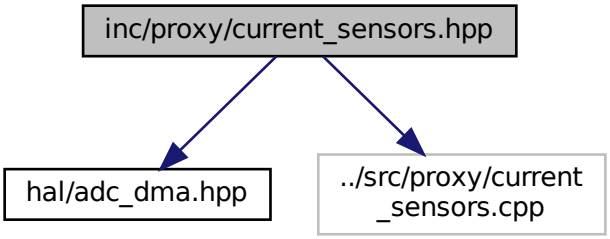
03/2024

8.18 inc/proxy/current_sensors.hpp File Reference

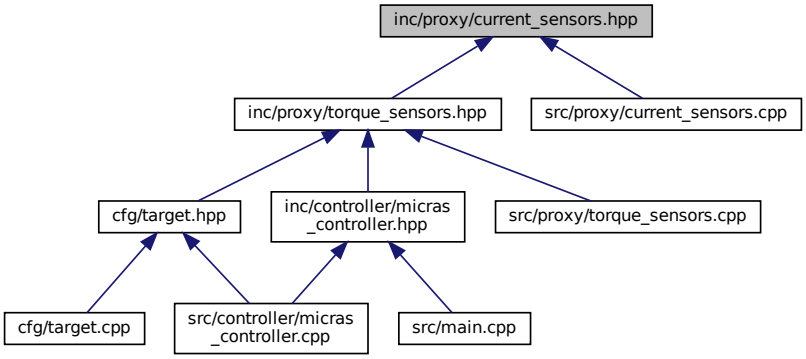
Proxy CurrentSensors class header.

```
#include "hal/adc_dma.hpp"
#include "../src/proxy/current_sensors.cpp"
```

Include dependency graph for current_sensors.hpp:



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



Classes

- class proxy::CurrentSensors< num_of_sensors >

  Class for controlling CurrentSensors.
- struct proxy::CurrentSensors< num_of_sensors >::Config

  Configuration structure for current sensors.

Namespaces

- proxy

8.18.1 Detailed Description

Proxy CurrentSensors class header.

Date

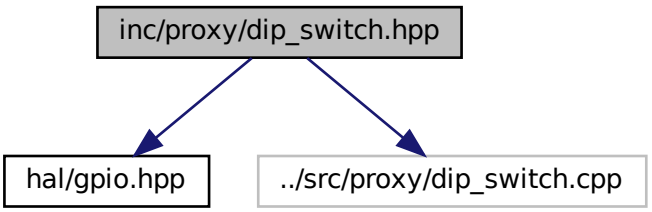
03/2024

8.19 inc/proxy/dip_switch.hpp File Reference

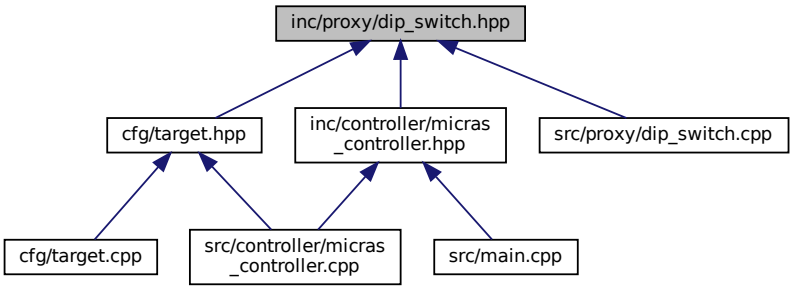
Proxy Dip Switch class header.

```
#include "hal/gpio.hpp"
#include "../src/proxy/dip_switch.cpp"
```

Include dependency graph for dip_switch.hpp:



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



Classes

- class proxy::DipSwitch< num_of_switches >

  Class for controlling a dip switch.
- struct proxy::DipSwitch< num_of_switches >::Config

  Configuration struct for DipSwitch.

Namespaces

- proxy

8.19.1 Detailed Description

Proxy Dip Switch class header.

Date

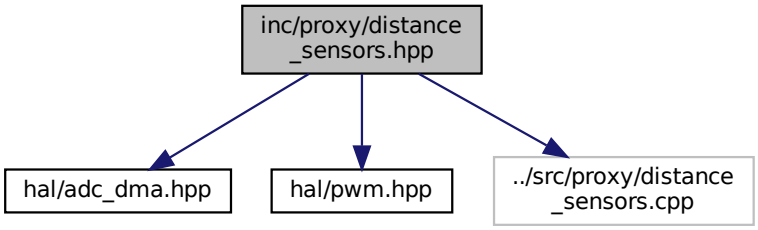
03/2024

8.20 inc/proxy/distance_sensors.hpp File Reference

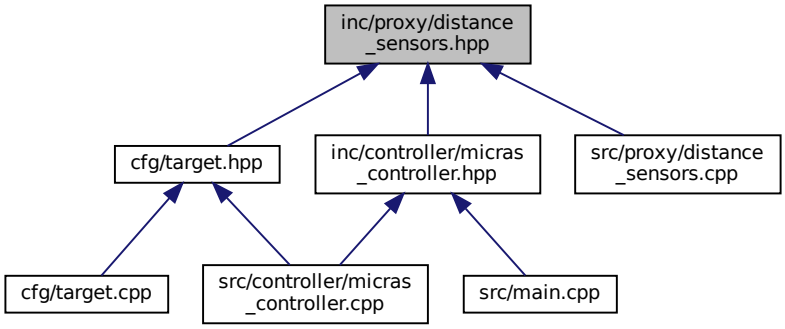
Proxy DistanceSensors class header.

```
#include "hal/adc_dma.hpp"
#include "hal/pwm.hpp"
#include "../src/proxy/distance_sensors.cpp"
```

Include dependency graph for distance_sensors.hpp:



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



Classes

- class proxy::DistanceSensors< num_of_sensors >

  Class for controlling DistanceSensors.
- struct proxy::DistanceSensors< num_of_sensors >::Config

  Configuration structure for distance sensors.

Namespaces

- proxy

8.20.1 Detailed Description

Proxy DistanceSensors class header.

Date

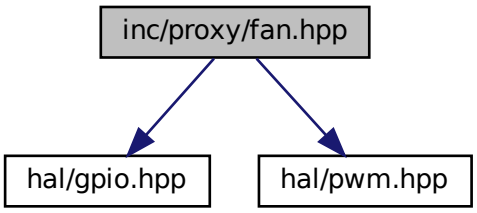
03/2024

8.21 inc/proxy/fan.hpp File Reference

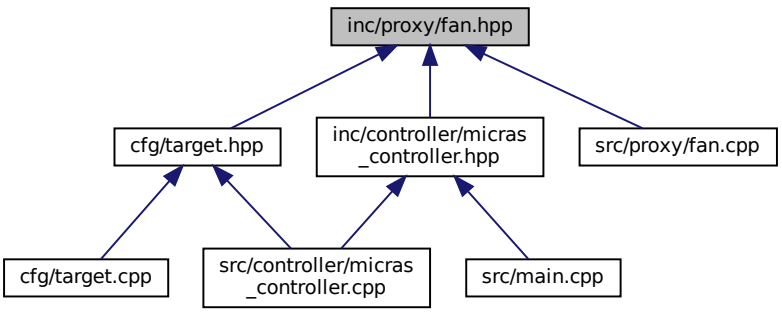
Proxy Fan class declaration.

```
#include "hal/gpio.hpp"
#include "hal/pwm.hpp"
```

Include dependency graph for fan.hpp:



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



Classes

- class proxy::Fan

  Class for controlling the fan driver.
- struct proxy::Fan::Config

  Configuration structure for the fan.

Namespaces

- proxy

8.21.1 Detailed Description

Proxy Fan class declaration.

Date

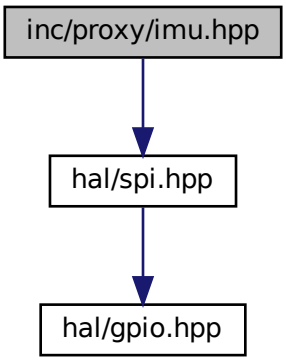
03/2024

8.22 inc/proxy/imu.hpp File Reference

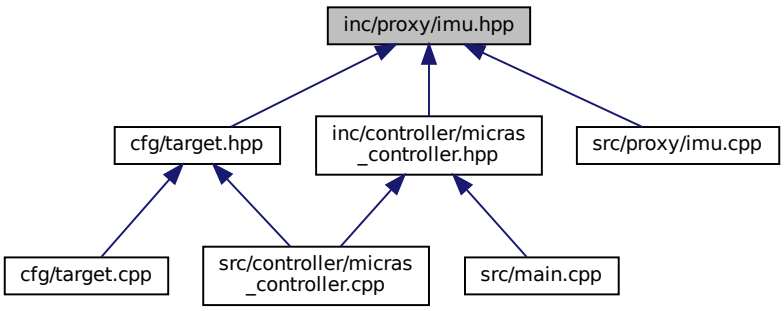
STM32 IMU HAL wrapper.

```
#include "hal/spi.hpp"
```

Include dependency graph for imu.hpp:



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



Classes

- class proxy::Imu

  Class to handle IMU peripheral on STM32 microcontrollers.
- struct proxy::Imu::Config

  IMU configuration struct.

Namespaces

- proxy

8.22.1 Detailed Description

STM32 IMU HAL wrapper.

Date

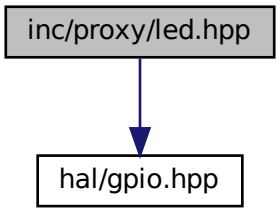
03/2024

8.23 inc/proxy/led.hpp File Reference

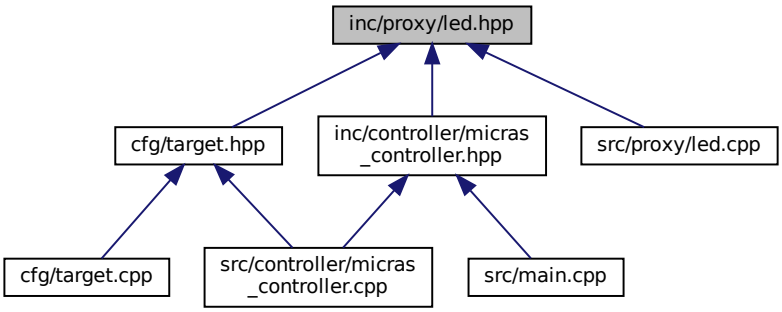
Proxy Led class header.

```
#include "hal/gpio.hpp"
```

Include dependency graph for led.hpp:



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



Classes

- class proxy::Led

  Class for controlling an LED.
- struct proxy::Led::Config

  Configuration structure for LED.

Namespaces

- proxy

8.23.1 Detailed Description

Proxy Led class header.

Date

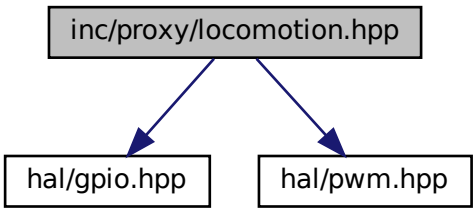
03/2024

8.24 inc/proxy/locomotion.hpp File Reference

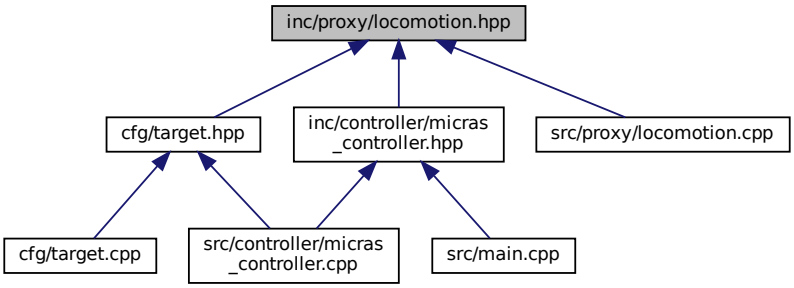
Proxy Locomotion class declaration.

```
#include "hal/gpio.hpp"
#include "hal/pwm.hpp"
```

Include dependency graph for locomotion.hpp:



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



Classes

- class proxy::Locomotion

  Class for controlling the locomotion driver.
- struct proxy::Locomotion::Config

  Configuration structure for the locomotion.

Namespaces

- proxy

8.24.1 Detailed Description

Proxy Locomotion class declaration.

Date

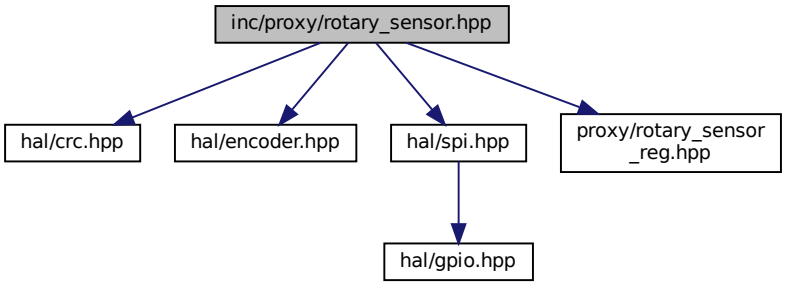
03/2024

8.25 inc/proxy/rotary_sensor.hpp File Reference

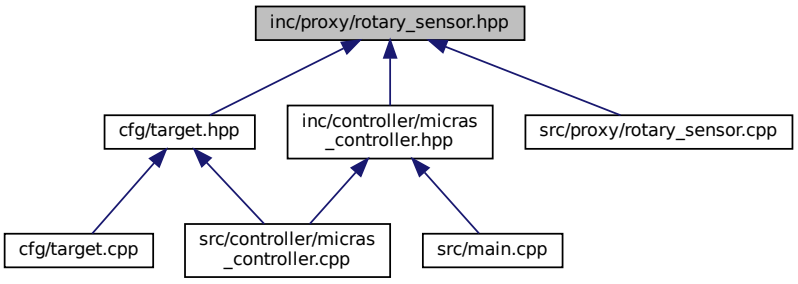
STM32 rotary sensor HAL wrapper.

```
#include "hal/crc.hpp"
#include "hal/encoder.hpp"
#include "hal/spi.hpp"
#include "proxy/rotary_sensor_reg.hpp"
```

Include dependency graph for rotary_sensor.hpp:



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



Classes

- class proxy::RotarySensor

  Class to handle rotary sensor peripheral on STM32 microcontrollers.
- struct proxy::RotarySensor::Config

  Rotary sensor configuration struct.
- struct proxy::RotarySensor::CommandFrame::Fields
- struct proxy::RotarySensor::DataFrame::Fields

Namespaces

- proxy

8.25.1 Detailed Description

STM32 rotary sensor HAL wrapper.

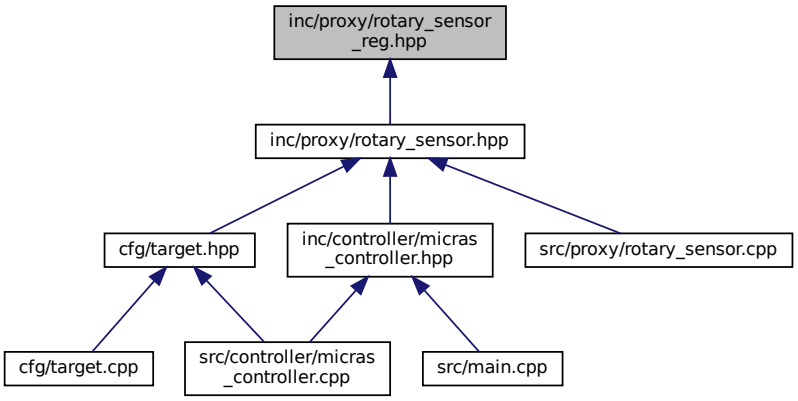
Date

03/2024

8.26 inc/proxy/rotary_sensor_reg.hpp File Reference

AS5047U rotary sensor registers definition.

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



Classes

- struct Registers

  Registers class for the rotary sensor configuration.
- union Registers::Disable

  Registers union types definition.
- struct Registers::Disable::Fields
- union Registers::Zposm
- struct Registers::Zposm::Fields
- union Registers::Zposl
- struct Registers::Zposl::Fields
- union Registers::Settings1
- struct Registers::Settings1::Fields
- union Registers::Settings2
- struct Registers::Settings2::Fields
- union Registers::Settings3
- struct Registers::Settings3::Fields
- union Registers::Ecc
- struct Registers::Ecc::Fields

8.26.1 Detailed Description

AS5047U rotary sensor registers definition.

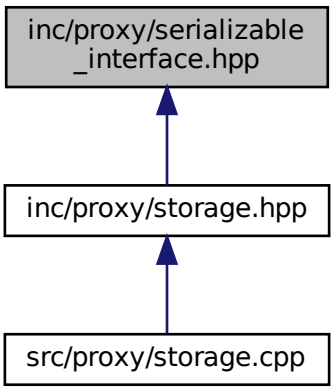
Date

03/2024

8.27 inc/proxy/serializable_interface.hpp File Reference

Serializable interface for all classes that need to be serialized.

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



Classes

- class ISerializable

  Interface class for serializable classes.

8.27.1 Detailed Description

Serializable interface for all classes that need to be serialized.

Date

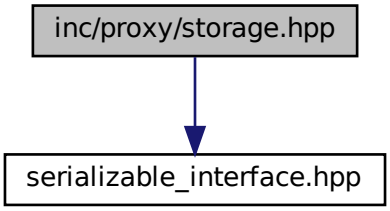
03/2024

8.28 inc/proxy/storage.hpp File Reference

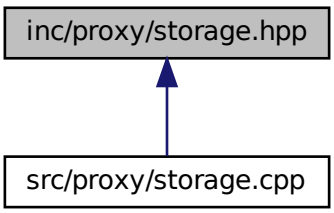
Proxy Storage class declaration.

```
#include "serializable_interface.hpp"
```

Include dependency graph for storage.hpp:



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



Classes

- class proxy::Storage

  Class for controlling the storage.
- struct proxy::Storage::Config

  Configuration structure for the storage.

Namespaces

- proxy

Variables

- template<typename T >
  concept proxy::Fundamental = std::is_fundamental<T>::value

8.28.1 Detailed Description

Proxy Storage class declaration.

Date

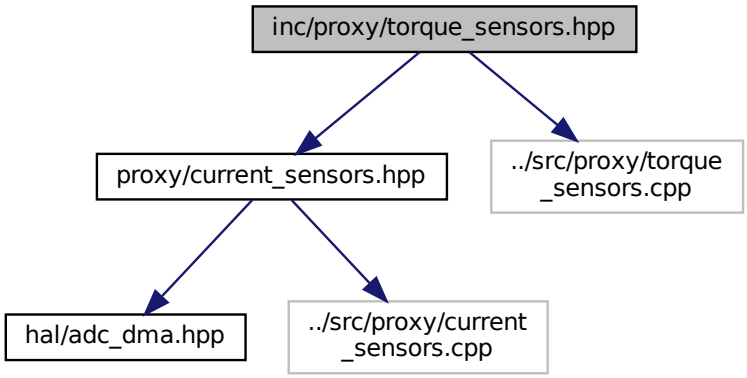
03/2024

8.29 inc/proxy/torque_sensors.hpp File Reference

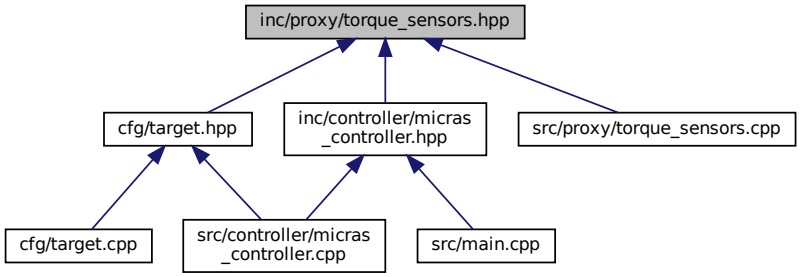
Proxy TorqueSensors class header.

```
#include "proxy/current_sensors.hpp"
#include "../src/proxy/torque_sensors.cpp"
```

Include dependency graph for torque_sensors.hpp:



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



Classes

- class proxy::TorqueSensors< num_of_sensors >

  Class for controlling TorqueSensors.
- struct proxy::TorqueSensors< num_of_sensors >::Config

  Configuration structure for torque sensors.

Namespaces

- proxy

8.29.1 Detailed Description

Proxy TorqueSensors class header.

Date

03/2024

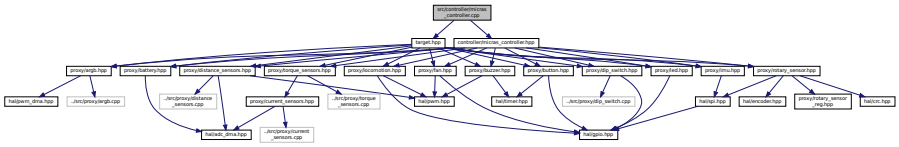
8.30 README.md File Reference

8.31 src/controller/micras_controller.cpp File Reference

Micras Controller class implementation.

```
#include "controller/micras_controller.hpp"
#include "target.hpp"
```

Include dependency graph for micras_controller.cpp:



8.31.1 Detailed Description

Micras Controller class implementation.

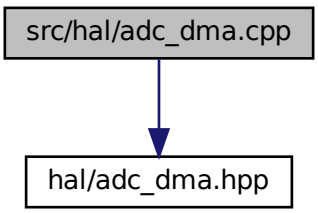
Date

03/2024

8.32 src/hal/adc_dma.cpp File Reference

STM32 ADC DMA HAL wrapper.

```
#include "hal/adc_dma.hpp"
```
Include dependency graph for adc_dma.cpp:



Namespaces

- hal

8.32.1 Detailed Description

STM32 ADC DMA HAL wrapper.

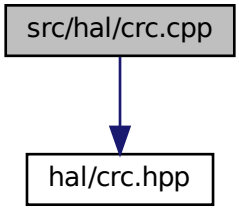
Date

03/2024

8.33 src/hal/crc.cpp File Reference

STM32 CRC HAL wrapper.

```
#include "hal/crc.hpp"
```
Include dependency graph for crc.cpp:



Namespaces

- hal

8.33.1 Detailed Description

STM32 CRC HAL wrapper.

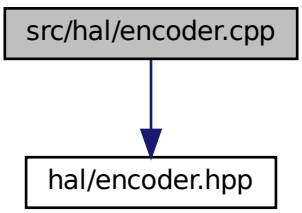
Date

03/2024

8.34 src/hal/encoder.cpp File Reference

STM32 encoder HAL wrapper.

```
#include "hal/encoder.hpp"
```
Include dependency graph for encoder.cpp:



Namespaces

- hal

8.34.1 Detailed Description

STM32 encoder HAL wrapper.

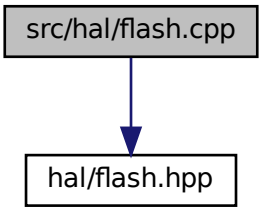
Date

03/2024

8.35 src/hal/flash.cpp File Reference

STM32 flash HAL wrapper.

```
#include "hal/flash.hpp"
```
Include dependency graph for flash.cpp:



Namespaces

- hal

8.35.1 Detailed Description

STM32 flash HAL wrapper.

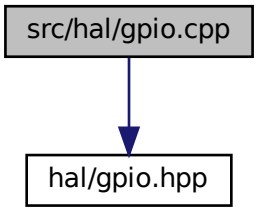
Date

03/2024

8.36 src/hal/gpio.cpp File Reference

HAL GPIO class source.

`#include "hal/gpio.hpp"`
Include dependency graph for gpio.cpp:



Namespaces

- hal

8.36.1 Detailed Description

HAL GPIO class source.

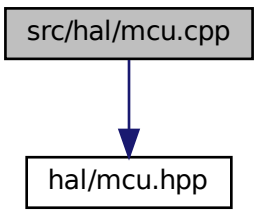
Date

03/2024

8.37 src/hal/mcu.cpp File Reference

MCU related.

`#include "hal/mcu.hpp"`
Include dependency graph for mcu.cpp:



Namespaces

- hal

Functions

- void SystemClock_Config ()
  Initializes System Clock.

8.37.1 Detailed Description

MCU related.

8.37.2 Function Documentation

8.37.2.1 SystemClock_Config()

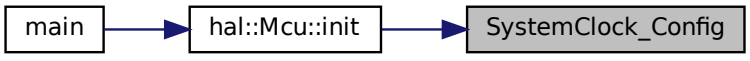
```
void SystemClock_Config ( )
```

Initializes System Clock.

Note

Defined by cube

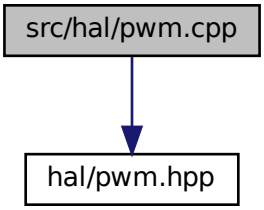
Here is the caller graph for this function:



8.38 src/hal/pwm.cpp File Reference

STM32 PWM HAL wrapper.

```
#include "hal/pwm.hpp"
```
Include dependency graph for pwm.cpp:



Namespaces

- hal

8.38.1 Detailed Description

STM32 PWM HAL wrapper.

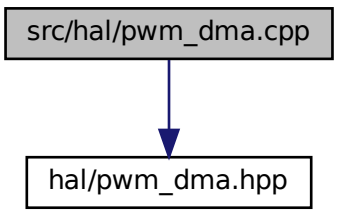
Date

03/2024

8.39 src/hal/pwm_dma.cpp File Reference

STM32 PWM DMA HAL wrapper.

```
#include "hal/pwm_dma.hpp"
```
Include dependency graph for pwm_dma.cpp:



Namespaces

- hal

8.39.1 Detailed Description

STM32 PWM DMA HAL wrapper.

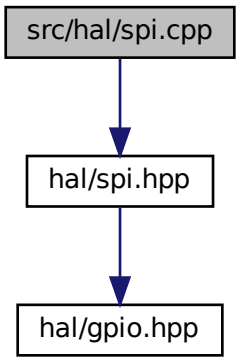
Date

03/2024

8.40 src/hal/spi.cpp File Reference

Proxy SPI Switch class source.

```
#include "hal/spi.hpp"
```
Include dependency graph for spi.cpp:



Namespaces

- hal

8.40.1 Detailed Description

Proxy SPI Switch class source.

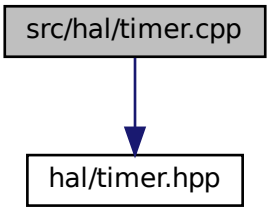
Date

03/2024

8.41 src/hal/timer.cpp File Reference

STM32 TIM HAL wrapper.

```
#include "hal/timer.hpp"
```
Include dependency graph for timer.cpp:



Namespaces

- hal

8.41.1 Detailed Description

STM32 TIM HAL wrapper.

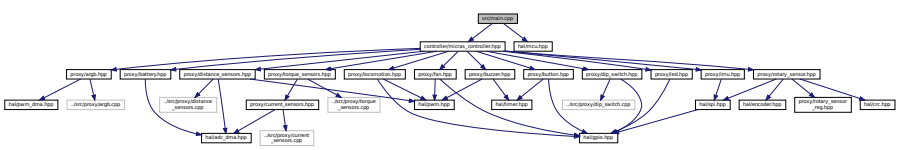
Date

03/2024

8.42 src/main.cpp File Reference

Main function.

```
#include "controller/micras_controller.hpp"
#include "hal/mcu.hpp"
```
Include dependency graph for main.cpp:



Functions

- int main ()

8.42.1 Detailed Description

Main function.

Date

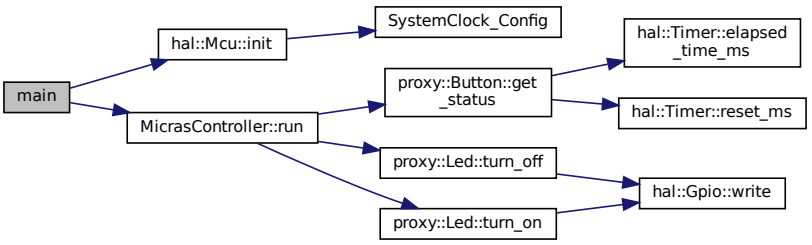
03/2024

8.42.2 Function Documentation

8.42.2.1 main()

```
int main ( )
```

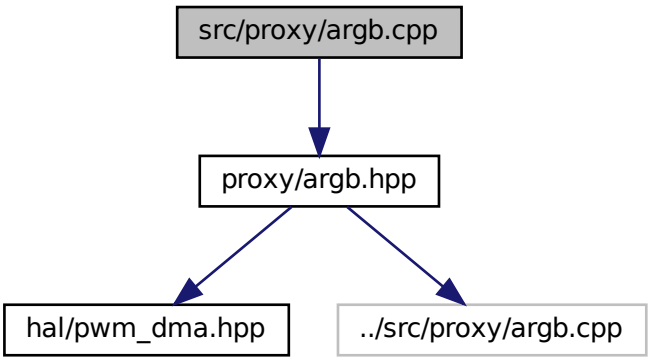
Here is the call graph for this function:



8.43 src/proxy/argb.cpp File Reference

Proxy Argb class implementation.

```
#include "proxy/argb.hpp"
```
Include dependency graph for argb.cpp:



Namespaces

- proxy

Macros

- #define MICRAS_PROXY_ARGB_CPP

8.43.1 Detailed Description

Proxy Argb class implementation.

Date

03/2024

8.43.2 Macro Definition Documentation

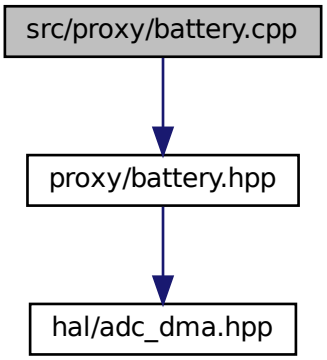
8.43.2.1 MICRAS_PROXY_ARGB_CPP

```
#define MICRAS_PROXY_ARGB_CPP
```

8.44 src/proxy/battery.cpp File Reference

Proxy Battery class implementation.

`#include "proxy/battery.hpp"`
Include dependency graph for battery.cpp:



Namespaces

- proxy

8.44.1 Detailed Description

Proxy Battery class implementation.

Date

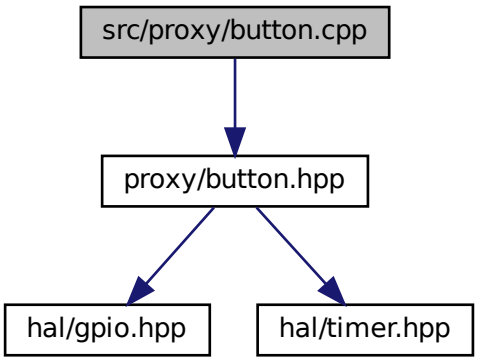
03/2024

8.45 src/proxy/button.cpp File Reference

Proxy Button class source.

```
#include "proxy/button.hpp"
```

Include dependency graph for button.cpp:



Namespaces

- proxy

8.45.1 Detailed Description

Proxy Button class source.

Date

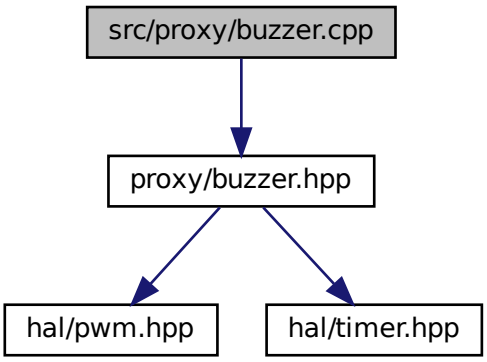
03/2024

8.46 src/proxy/buzzer.cpp File Reference

Proxy Buzzer class implementation.

```
#include "proxy/buzzer.hpp"
```

Include dependency graph for buzzer.cpp:



Namespaces

- proxy

8.46.1 Detailed Description

Proxy Buzzer class implementation.

Date

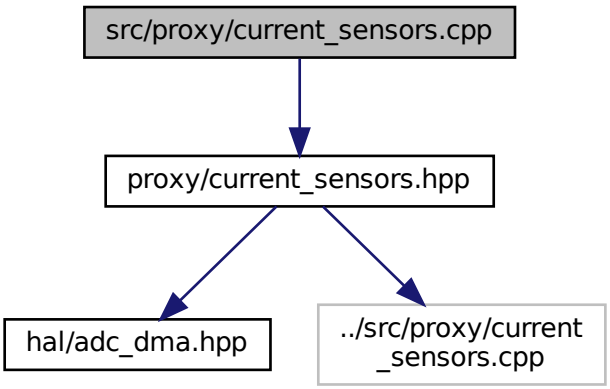
03/2024

8.47 src/proxy/current_sensors.cpp File Reference

Proxy CurrentSensors class implementation.

```
#include "proxy/current_sensors.hpp"
```

Include dependency graph for current_sensors.cpp:



Namespaces

- proxy

Macros

- #define MICRAS_PROXY_CURRENT_SENSORS_CPP

8.47.1 Detailed Description

Proxy CurrentSensors class implementation.

Date

03/2024

8.47.2 Macro Definition Documentation

8.47.2.1 MICRAS_PROXY_CURRENT_SENSORS_CPP

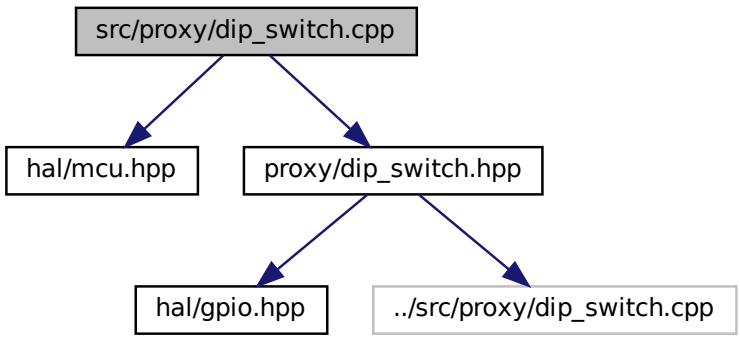
```
#define MICRAS_PROXY_CURRENT_SENSORS_CPP
```

8.48 src/proxy/dip_switch.cpp File Reference

Proxy DIP Switch class source.

```
#include "hal/mcu.hpp"
#include "proxy/dip_switch.hpp"
```

Include dependency graph for dip_switch.cpp:



Namespaces

- proxy

Macros

- #define MICRAS_PROXY_DIP_SWITCH_CPP

8.48.1 Detailed Description

Proxy DIP Switch class source.

Date

03/2024

8.48.2 Macro Definition Documentation

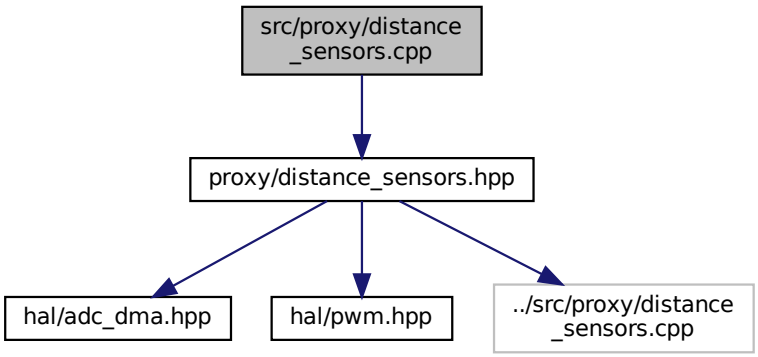
8.48.2.1 MICRAS_PROXY_DIP_SWITCH_CPP

```
#define MICRAS_PROXY_DIP_SWITCH_CPP
```

8.49 src/proxy/distance_sensors.cpp File Reference

```
#include "proxy/distance_sensors.hpp"
```

Include dependency graph for distance_sensors.cpp:



Namespaces

- proxy

Macros

- #define MICRAS_PROXY_DISTANCE_SENSORS_CPP

8.49.1 Macro Definition Documentation

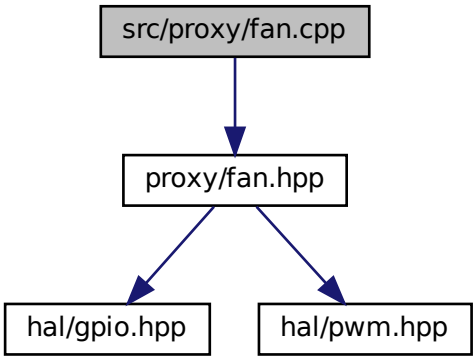
8.49.1.1 MICRAS_PROXY_DISTANCE_SENSORS_CPP

```
#define MICRAS_PROXY_DISTANCE_SENSORS_CPP
```

8.50 src/proxy/fan.cpp File Reference

Proxy Fan class source.

```
#include "proxy/fan.hpp"
```
Include dependency graph for fan.cpp:



Namespaces

- proxy

8.50.1 Detailed Description

Proxy Fan class source.

Date

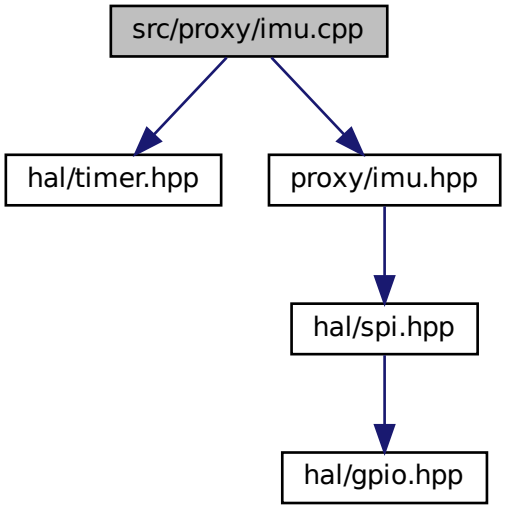
03/2024

8.51 src/proxy/imu.cpp File Reference

Proxy Imu class source.

```
#include "hal/timer.hpp"
#include "proxy/imu.hpp"
```

Include dependency graph for imu.cpp:



Namespaces

- proxy

8.51.1 Detailed Description

Proxy Imu class source.

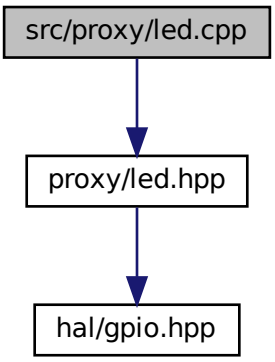
Date

03/2024

8.52 src/proxy/led.cpp File Reference

Proxy Led class source.

```
#include "proxy/led.hpp"
```
Include dependency graph for led.cpp:



Namespaces

- proxy

8.52.1 Detailed Description

Proxy Led class source.

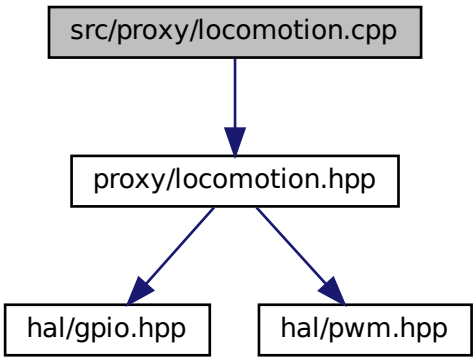
Date

03/2024

8.53 src/proxy/locomotion.cpp File Reference

Proxy Locomotion class source.

```
#include "proxy/locomotion.hpp"
```
Include dependency graph for locomotion.cpp:



Namespaces

- proxy

8.53.1 Detailed Description

Proxy Locomotion class source.

Date

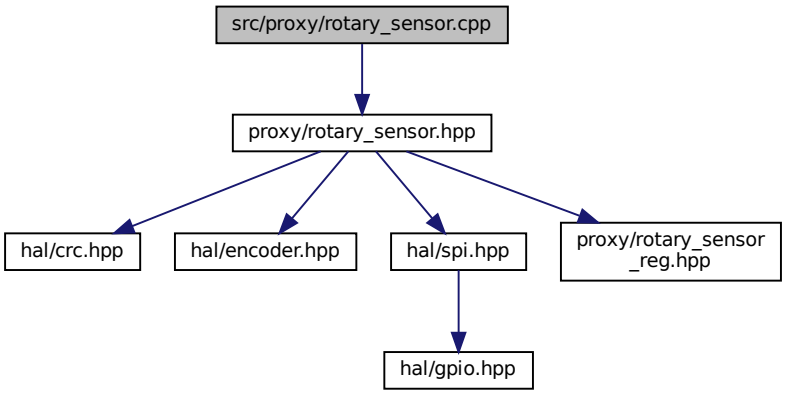
03/2024

8.54 src/proxy/rotary_sensor.cpp File Reference

Proxy RotarySensor class source.

```
#include "proxy/rotary_sensor.hpp"
```

Include dependency graph for rotary_sensor.cpp:



Namespaces

- proxy

8.54.1 Detailed Description

Proxy RotarySensor class source.

Date

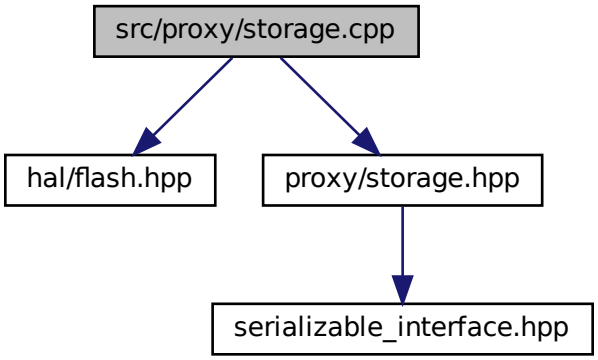
03/2024

8.55 src/proxy/storage.cpp File Reference

Proxy Storage class source.

```
#include "hal/flash.hpp"
#include "proxy/storage.hpp"
```

Include dependency graph for storage.cpp:



Namespaces

- proxy

8.55.1 Detailed Description

Proxy Storage class source.

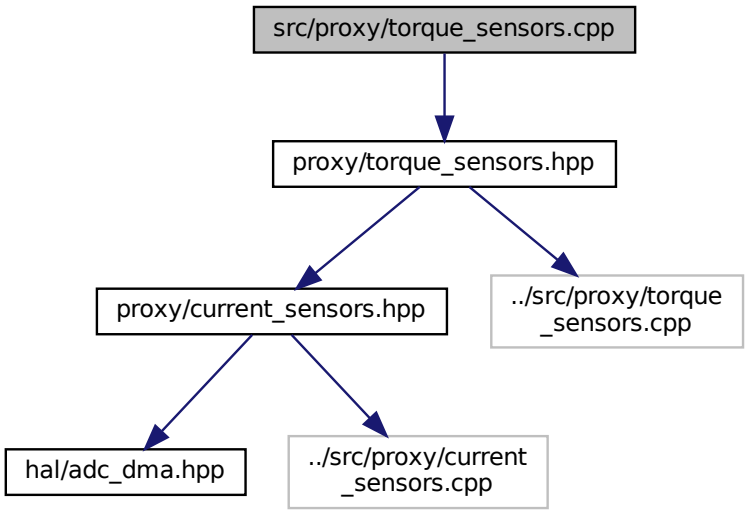
Date

03/2024

8.56 src/proxy/torque_sensors.cpp File Reference

Proxy TorqueSensors class implementation.

```
#include "proxy/torque_sensors.hpp"
```
Include dependency graph for torque_sensors.cpp:



Namespaces

- proxy

Macros

- #define MICRAS_PROXY_TORQUE_SENSORS_CPP

8.56.1 Detailed Description

Proxy TorqueSensors class implementation.

Date

03/2024

8.56.2 Macro Definition Documentation

8.56.2.1 MICRAS_PROXY_TORQUE_SENSORS_CPP

```
#define MICRAS_PROXY_TORQUE_SENSORS_CPP
```

Index