-2233

LIS3MDLTR Driver

0.1

Generated by Doxygen 1.9.1

Contents

Chapter 1

Home Page

1.1 Introduction

This program, written in C, is a driver for LIS3MDLTR - Digital output magnetic sensor (ultralow-power, high-performance 3-axis magnetometer). It follows an object-oriented style and implements a two-layer architecture consisting of HAL (Hardware Abstraction Layer) and PAL (Platform Abstraction Layer). The program utilizes the Unity/Ceedling framework for unit testing.

This Driver implemented below purposes:

1. Retrieve the device's full-scale configuration
2. Retrieve and set the device's output data rate
3. Enable or disable the device's interrupt pin
4. Read the output data of a specific axis

Chapter 2

Module Index

2.1 Modules

Here is a list of all modules:

LIS3MDLTR . . . . . **??**

Chapter 3

Class Index

3.1 Class List

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

LIS3MDLTR
Represents the configuration and state of a LIS3MDLTR sensor . . . . . **??**

Chapter 4

Module Documentation

4.1 LIS3MDLTR

Driver module for LIS3MDLTR 3-Axis Magnetometer sensor.

Classes

- class LIS3MDLTR

  Represents the configuration and state of a LIS3MDLTR sensor.

Macros

- #define X_ADDRESS_REG ((uint8_t)0x28)

  Address of X register.
- #define Y_ADDRESS_REG ((uint8_t)0x2A)

  Address of Y register.
- #define Z_ADDREDD_REG ((uint8_t)0x2C)

  Address of Z register.
- #define DEFAULT_ADDRESS ((uint8_t)0x3d)

  Default bus address.

Typedefs

- typedef enum FunctionStatus **FunctionStatus**
- typedef enum full_scale_t **full_scale_t**
- typedef enum data_rate_t **data_rate_t**
- typedef enum interupt_status_t **interupt_status_t**
- typedef enum axis_t **axis_t**

Enumerations

- enum FunctionStatus {
  FUNCTION_STATUS_OK = 0 , **FUNCTION_STATUS_ARGUMENT_ERROR** , **FUNCTION_STATUS_↩ DEVICE_NOT_INTIALIZED** , **FUNCTION_STATUS_BOUNDARY_ERROR** ,
  FUNCTION_STATUS_ERROR }

  Enumeration representing the status of a function.
- enum full_scale_t { SCLAE_4G = 0x00 , SCLAE_8G = 0x01 , SCALE_12G = 0x02 , SCALE_16G = 0x03 }
- enum data_rate_t {
  RATE_0_625 = 0 , RATE_1_25 , RATE_2_5 , RATE_5 ,
  RATE_10 , RATE_20 , RATE_40 , RATE_80 ,
  RATE_LP , RATE_MP , RATE_HP , RATE_UHP ,
  RATE_ENUM_END }

  Enumeration representing different data rates for a sensor.
- enum **interupt_status_t** { **DISABLE** = 0 , **ENABLE** = 1 }
- enum **axis_t** { **AXIS_X** = X_ADDRESS_REG , **AXIS_Y** = Y_ADDRESS_REG , **AXIS_Z** = Z_ADDREDD_↩ REG }

Functions

- FunctionStatus LIS3MDLTR_Constructor (struct LIS3MDLTR ∗self_ptr, uint8_t devise_id)

  Initializes the LIS3MDLTR sensor object.
- FunctionStatus LIS3MDLTR_GetFullScaleConfig (struct LIS3MDLTR ∗self_ptr, full_scale_t ∗fullscale)

  Gets the full-scale configuration setting from the LIS3MDLTR sensor.
- FunctionStatus LIS3MDLTR_ChangeOutputDataRate (struct LIS3MDLTR ∗self_ptr, data_rate_t rate)

  Changes the output data rate setting on the LIS3MDLTR sensor.
- FunctionStatus LIS3MDLTR_GetOutputDataRate (struct LIS3MDLTR ∗self_ptr, data_rate_t ∗datarate)

  Gets the output data rate setting from the LIS3MDLTR sensor.
- FunctionStatus LIS3MDLTR_ChangeInteruptPinStatus (struct LIS3MDLTR ∗self_ptr, interupt_status_↩ t intrupt_status)

  Changes the status of the interrupt pin on the LIS3MDLTR sensor.
- FunctionStatus LIS3MDLTR_ReadAxis (struct LIS3MDLTR ∗self_ptr, axis_t axis_to_read, uint16_t ∗data_ptr)

  Reads magnetic field data for a specified axis from the LIS3MDLTR sensor.

4.1.1 Detailed Description

Driver module for LIS3MDLTR 3-Axis Magnetometer sensor.

Author

: Parvin Parsa

Date

: 10.Dec.2023 Github : itriplep LIS3MDLTR implementation

This is the driver module for LIS3MDLTR 3-Axis magnetometer sensor It uses the I2C communication to get 3-Axis and Also get/set registers of sensors.

4.1.2 Enumeration Type Documentation

4.1.2.1 data_rate_t

`enum data_rate_t`

Enumeration representing different data rates for a sensor.

This enumeration defines various data rates ranging from 0.625 to 1000 Hz, with additional symbolic values for specific rates (LP, MP, HP, UHP). RATE_ENUM_END is included to mark the end of the data rate enumeration.

Enumerator

RATE_0_625	Data rate 0.625.
RATE_1_25	Data rate 1.25.
RATE_2_5	Data rate 2.5.
RATE_5	Data rate 5.
RATE_10	Data rate 10.
RATE_20	Data rate 20.
RATE_40	Data rate 40.
RATE_80	Data rate 80.
RATE_LP	Data rate 1000.
RATE_MP	Data rate 560.
RATE_HP	Data rate 300.
RATE_UHP	Data rate 155.
RATE_ENUM_END	End of data rate.

Definition at line 110 of file LIS3MDLTR.h.

4.1.2.2 full_scale_t

`enum full_scale_t`

Enumerator

SCLAE_4G	+/-4 gauss scale
SCLAE_8G	+/-8 gauss scale
SCALE_12G	+/-12 gauss scale
SCALE_16G	+/-16 gauss scale

Definition at line 95 of file LIS3MDLTR.h.

4.1.2.3 FunctionStatus

```
enum FunctionStatus
```

Enumeration representing the status of a function.

This enumeration defines different status codes that a function may return, including success (OK), argument error, uninitialized device, boundry error, and general error.

Definition at line 81 of file LIS3MDLTR.h.

4.1.3 Function Documentation

4.1.3.1 LIS3MDLTR_ChangeInteruptPinStatus()

```
FunctionStatus LIS3MDLTR_ChangeInteruptPinStatus (
            struct LIS3MDLTR * self_ptr,
            interupt_status_t intrupt_status )
```

Changes the status of the interrupt pin on the LIS3MDLTR sensor.

LIS3MDLTR_ChangeIntruptPinStatus

Detailed Description

Parameters

self_ptr	Pointer to LIS3MDLTR structure to read data from
intrupt_status	Desired status of the interrupt pin (ENABLE or DISABLE)

Returns

FunctionStatus

- FUNCTION_STATUS_OK if the operation was successful
- FUNCTION_STATUS_ARGUMENT_ERROR if self_ptr is NULL
- FUNCTION_STATUS_DEVICE_NOT_INTIALIZED if the sensor is not initialized
- FUNCTION_STATUS_BOUNDARY_ERROR if intrupt_status is not ENABLE or DISABLE

This function changes the status of the interrupt pin on the LIS3MDLTR sensor. It performs necessary argument checks, reads the current interrupt configuration(from INT_CFG register), updates the desired status, and writes the new configuration back to the sensor using I2C communication.

Definition at line 193 of file LIS3MDLTR.c.

4.1.3.2 LIS3MDLTR_ChangeOutputDataRate()

```
FunctionStatus LIS3MDLTR_ChangeOutputDataRate (
            struct LIS3MDLTR * self_ptr,
            data_rate_t rate )
```

Changes the output data rate setting on the LIS3MDLTR sensor.
LIS3MDLTR_ChangeOutputDataRate
Detailed Description

Parameters

self_ptr	Pointer to LIS3MDLTR structure to read data from
rate	Desired data rate setting (data_rate_t enum)

Returns

FunctionStatus

- FUNCTION_STATUS_OK if the operation was successful
- FUNCTION_STATUS_ARGUMENT_ERROR if self_ptr is NULL
- FUNCTION_STATUS_DEVICE_NOT_INTIALIZED if the sensor is not initialized
- FUNCTION_STATUS_BOUNDARY_ERROR if the rate is out of bounds

This function changes the output data rate setting on the LIS3MDLTR sensor. It performs necessary argument checks, reads the current rate configuration(from CTRL_REG1 register), updates the desired rate, and writes the new configuration back to the sensor using I2C communication. The new rate is determined based on the provided rate parameter and sensor configuration.

Definition at line 132 of file LIS3MDLTR.c.

4.1.3.3 LIS3MDLTR_Constructor()

```
FunctionStatus LIS3MDLTR_Constructor (
            struct LIS3MDLTR * self_ptr,
            uint8_t devise_id )
```

Initializes the LIS3MDLTR sensor object.
LIS3MDLTR_Constructor
Detailed Description

Parameters

self_ptr	Pointer to the LIS3MDLTR structure to be initialized
device_id	Device identifier for the LIS3MDLTR sensor

Returns

FunctionStatus

- FUNCTION_STATUS_OK if the operation was successful
- FUNCTION_STATUS_ARGUMENT_ERROR if self_ptr is NULL
- Other error codes can be defined based on the specific implementation

This function initializes the LIS3MDLTR sensor object. It sets the device identifier, and if a valid device_id is provided, it uses it; otherwise, it defaults to the DEFAULT_ADDRESS. The initialization flag is set to true, indicating that the sensor object has been properly initialized.

Definition at line 93 of file LIS3MDLTR.c.

4.1.3.4 LIS3MDLTR_GetFullScaleConfig()

```
FunctionStatus LIS3MDLTR_GetFullScaleConfig (
            struct LIS3MDLTR * self_ptr,
            full_scale_t * fullscale )
```

Gets the full-scale configuration setting from the LIS3MDLTR sensor.
LIS3MDLTR_GetFullScaleConfig
Detailed Description

Parameters

self_ptr	Pointer to the LIS3MDLTR structure to read data from
fullscale	Pointer to a full_scale_t variable where the full-scale setting will be stored

Returns

FunctionStatus

- FUNCTION_STATUS_OK if the operation was successful
- FUNCTION_STATUS_ARGUMENT_ERROR if self_ptr is NULL
- FUNCTION_STATUS_DEVICE_NOT_INTIALIZED if the sensor is not initialized

This function retrieves the full-scale configuration setting from the LIS3MDLTR sensor. It performs necessary argument checks, reads the control register(from CTRL_REG2 register), and extracts the full-scale information based on the sensor's configuration. The result is stored in the provided fullscale variable.

Definition at line 110 of file LIS3MDLTR.c.

4.1.3.5 LIS3MDLTR_GetOutputDataRate()

```
FunctionStatus LIS3MDLTR_GetOutputDataRate (
            struct LIS3MDLTR * self_ptr,
            data_rate_t * datarate )
```

Gets the output data rate setting from the LIS3MDLTR sensor.
LIS3MDLTR_GetOutputDataRate
Detailed Description

Parameters

self_ptr	Pointer to LIS3MDLTR structure to read data from
datarate	Pointer to a data_rate_t variable where the data rate setting will be stored

Returns

FunctionStatus

- FUNCTION_STATUS_OK if the operation was successful
- FUNCTION_STATUS_ARGUMENT_ERROR if self_ptr is NULL
- FUNCTION_STATUS_DEVICE_NOT_INTIALIZED if the sensor is not initialized

This function retrieves the output data rate setting from the LIS3MDLTR sensor. It performs necessary argument checks, reads the control register(from CTRL_REG1 register), and extracts the data rate information based on the sensor's configuration. The result is stored in the provided datarate variable.

Definition at line 168 of file LIS3MDLTR.c.

4.1.3.6 LIS3MDLTR_ReadAxis()

```
FunctionStatus LIS3MDLTR_ReadAxis (
            struct LIS3MDLTR * self_ptr,
            axis_t axis_to_read,
            uint16_t * data_ptr )
```

Reads magnetic field data for a specified axis from the LIS3MDLTR sensor.

LIS3MDLTR_ReadAxis
Detailed Description

Parameters

self_ptr	Pointer to LIS3MDLTR structure to read data from
axis_to_read	The axis for which data needs to be read (AXIS_X, AXIS_Y, or AXIS_Z)
data_ptr	Pointer to a uint16_t variable where the read data will be stored

Returns

FunctionStatus

- FUNCTION_STATUS_OK if the operation was successful
- FUNCTION_STATUS_ARGUMENT_ERROR if self_ptr is NULL
- FUNCTION_STATUS_DEVICE_NOT_INTIALIZED if the sensor is not initialized
- FUNCTION_STATUS_BOUNDARY_ERROR if the axis_to_read is out of bounds (not within AXIS_X to AXIS_Z)

This function reads magnetic field data from the LIS3MDLTR sensor for a specified axis. It performs necessary argument checks, initializes a buffer to store the read data, and reads the data from the sensor using I2C communication. The result is stored in the provided data_ptr.

Definition at line 223 of file LIS3MDLTR.c.

Chapter 5

Class Documentation

5.1 LIS3MDLTR Class Reference

Represents the configuration and state of a LIS3MDLTR sensor.
`#include <LIS3MDLTR.h>`

Public Attributes

- bool **initialization**
- uint8_t **device_id**

5.1.1 Detailed Description

Represents the configuration and state of a LIS3MDLTR sensor.
DETAILED DESCRIPTION
\members

- `initialization`: A boolean indicating whether the sensor is initialized.
- `device_id`: The bus address of the sensor.

Definition at line 69 of file LIS3MDLTR.h.
The documentation for this class was generated from the following file:

- platform/inc/LIS3MDLTR.h