AdafruitLIS3MDL Library Documentation

Release 1.0

Bryan Siepert

Jun 07, 2021

Contents

CircuitPython helper library for the LIS3MDL 3-axis magnetometer

CHAPTER 1

Dependencies

This driver depends on:

- Adafruit CircuitPython
- Bus Device
- Register

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the Adafruit library and driver bundle.

CHAPTER 2

Installing from PyPI

On supported GNU/Linux systems like the Raspberry Pi, you can install the driver locally from PyPI. To install for current user:

```
pip3 install adafruit-circuitpython-lis3mdl
```

To install system-wide (this may be required in some cases):

```
sudo pip3 install adafruit-circuitpython-lis3mdl
```

To install in a virtual environment in your current project:

```
mkdir project-name && cd project-name
python3 -m venv .env
source .env/bin/activate
pip3 install adafruit-circuitpython-lis3mdl
```

CHAPTER 3

Usage Example

```
import time
import board
import adafruit_lis3mdl

i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = adafruit_lis3mdl.LIS3MDL(i2c)

while True:
    mag_x, mag_y, mag_z = sensor.magnetic

    print('X:{0:10.2f}, Y:{1:10.2f}, Z:{2:10.2f} uT'.format(mag_x, mag_y, mag_z))
    print('')
    time.sleep(1.0)
```

CHAPTER 4

Contributing

Contributions are welcome! Please read our Code of Conduct before contributing to help this project stay welcoming.

CHAPTER 5

Documentation

For information on building library documentation, please check out this guide.

CHAPTER 6

Table of Contents

6.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/lis3mdl_simpletest.py

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

""" Display magnetometer data once per second """

import time
import board
import adafruit_lis3mdl

i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = adafruit_lis3mdl.LIS3MDL(i2c)

while True:
    mag_x, mag_y, mag_z = sensor.magnetic

    print("X:{0:10.2f}, Y:{1:10.2f}, Z:{2:10.2f} uT".format(mag_x, mag_y, mag_z))
    print("")
    time.sleep(1.0)
```

6.2 Compass Example

Use the magnetometer to calculate compass headings.

Listing 2: examples/lis3mdl_compass.py

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

""" Display compass heading data five times per second """
import time
from math import atan2, degrees
import board
import adafruit_lis3mdl

i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = adafruit_lis3mdl.LIS3MDL(i2c)


def vector_2_degrees(x, y):
    angle = degrees(atan2(y, x))
    if angle < 0:
        angle += 360
    return angle


def get_heading(_sensor):
    magnet_x, magnet_y, _ = _sensor.magnetic
    return vector_2_degrees(magnet_x, magnet_y)


while True:
    print("heading: {:.2f} degrees".format(get_heading(sensor)))
    time.sleep(0.2)
```

6.3 Data Rate Example

Test each data rate

Listing 3: examples/lis3mdl_data_rate_test.py

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

""" Test Each Data Rate """

# pylint: disable=no-member
import time
import board
from adafruit_lis3mdl import LIS3MDL, Rate, PerformanceMode

i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = LIS3MDL(i2c)

current_rate = Rate.RATE_155_HZ
sensor.data_rate = current_rate
start_time = time.monotonic()
print("data_rate is", Rate.string[sensor.data_rate], "HZ")
print("performance_mode is", PerformanceMode.string[sensor.performance_mode])
```

(continues on next page)

(continued from previous page)

```
while True:
    mag_x, mag_y, mag_z = sensor.magnetic

    print("X:{0:10.2f}, Y:{1:10.2f}, Z:{2:10.2f} uT".format(mag_x, mag_y, mag_z))

    # sleep for enough time so that we'll read the value twice per measurement
    sleep_time = 1 / (Rate.string[current_rate] * 2)
    time.sleep(sleep_time)

    # exit loop after a second to prevent hard to stop loops with short delays
    if (time.monotonic() - start_time) > 1:
        break
```

6.4 LSM6DS Test

Test the LSM6DS device

Listing 4: examples/lis3mdl_lsm6ds_test.py

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

import time
import board
from adafruit_lsm6ds.lsm6dsox import LSM6DSOX as LSM6DS

# To use LSM6DS33, comment out the LSM6DSOX import line
# and uncomment the next line
# from adafruit_lsm6ds.lsm6ds33 import LSM6DS33 as LSM6DS

# To use ISM330DHCX, comment out the LSM6DSOX import line
# and uncomment the next line
# from adafruit_lsm6ds.lsm330dhcx import ISM330DHCX as LSM6DS

from adafruit_lis3mdl import LIS3MDL

i2c = board.I2C()  # uses board.SCL and board.SDA
accel_gyro = LSM6DS(i2c)
mag = LIS3MDL(i2c)

while True:
    acceleration = accel_gyro.acceleration
    gyro = accel_gyro.gyro
    magnetic = mag.magnetic
    print(
        "Acceleration: X:{0:7.2f}, Y:{1:7.2f}, Z:{2:7.2f} m/s^2".format(*acceleration)
    )
    print("Gyro          X:{0:7.2f}, Y:{1:7.2f}, Z:{2:7.2f} rad/s".format(*gyro))
    print("Magnetic      X:{0:7.2f}, Y:{1:7.2f}, Z:{2:7.2f} uT".format(*magnetic))
    print("")
    time.sleep(0.5)
```

6.5 Range Test

Test each range

Listing 5: examples/lis3mdl_range_test.py

```
# SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
# SPDX-License-Identifier: MIT

""" Test Each range """
# pylint: disable=no-member
import time
import board
from adafruit_lis3mdl import LIS3MDL, Range

i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = LIS3MDL(i2c)

while True:

    for mag_range in [
        Range.RANGE_4_GAUSS,
        Range.RANGE_8_GAUSS,
        Range.RANGE_12_GAUSS,
        Range.RANGE_16_GAUSS,
    ]:
        sensor.range = mag_range
        print("Range: %d Gauss" % Range.string[sensor.range])
        mag_x, mag_y, mag_z = sensor.magnetic

        print("X:{0:10.2f}, Y:{1:10.2f}, Z:{2:10.2f} uT".format(mag_x, mag_y, mag_z))
        print("")
        time.sleep(0.3)
```

6.6 adafruit_lis3mdl

CircuitPython helper library for the LIS3MDL 3-axis magnetometer

- Author(s): Bryan Siepert

6.6.1 Implementation Notes

Hardware: * Adafruit Adafruit LSM6DS33 + LIS3MDL - 9 DoF IMU (Product ID: 4485)

Software and Dependencies:

- Adafruit CircuitPython firmware for the supported boards: https://circuitpython.org/downloads
- Adafruit's Bus Device library: https://github.com/adafruit/Adafruit_CircuitPython_BusDevice
- Adafruit's Register library: https://github.com/adafruit/Adafruit_CircuitPython_Register

class adafruit_lis3mdl.**LIS3MDL**(*i2c_bus*, *address=<sphinx.ext.autodoc.importer._MockObject object>*)

Driver for the LIS3MDL 3-axis magnetometer.

Parameters

- **i2c_bus** (*I2C*) – The I2C bus the LIS3MDL is connected to.
- **address** – The I2C device address. Defaults to `0x1C`

Quickstart: Importing and using the device

Here is an example of using the *LIS3MDL* class. First you will need to import the libraries to use the sensor

```
import board
import adafruit_lis3mdl
```

Once this is done you can define your `board.I2C` object and define your sensor object

```
i2c = board.I2C()
sensor = adafruit_lis3mdl.LIS3MDL(i2c)
```

Now you have access to the *magnetic* attribute

```
mag_x, mag_y, mag_z = sensor.magnetic
```

data_rate
The rate at which the sensor takes measurements. Must be a `Rate`

magnetic
The processed magnetometer sensor values. A 3-tuple of X, Y, Z axis values in microteslas that are signed floats.

operation_mode
The operating mode for the sensor, controlling how measurements are taken. Must be an *OperationMode*. See the the *OperationMode* document for additional details

performance_mode
Sets the 'performance mode' of the sensor. Must be a `PerformanceMode`. Note that *performance_mode* affects the available data rate and will be automatically changed by setting `data_rate` to certain values.

range
The measurement range for the magnetic sensor. Must be a `Range`

reset()
Reset the sensor to the default state set by the library

class adafruit_lis3mdl.**OperationMode**
Options for *operation_mode*

Operation Mode	Meaning
`OperationMode.CONTINUOUS`	Measurements are made continuously at the given *data_rate*
`OperationMode.SINGLE`	Setting to `SINGLE` takes a single measurement.
`OperationMode.POWER_DOWN`	Halts measurements. *magnetic* will return the last measurement

class adafruit_lis3mdl.**Rate**
Options for *data_rate*

Rate	Meaning
RATE_0_625_HZ	0.625 HZ
RATE_1_25_HZ	1.25 HZ
RATE_2_5_HZ	2.5 HZ
RATE_5_HZ	5 HZ
RATE_10_HZ	10 HZ
RATE_20_HZ	20 HZ
RATE_40_HZ	40 HZ
RATE_80_HZ	80 HZ
RATE_155_HZ	155 HZ (Sets PerformanceMode to MODE_ULTRA)
RATE_300_HZ	300 HZ (Sets PerformanceMode to MODE_HIGH)
RATE_560_HZ	560 HZ (Sets PerformanceMode to MODE_MEDIUM)
RATE_1000_HZ	1000 HZ (Sets PerformanceMode to MODE_LOW_POWER)

CHAPTER 7

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index