Adafruit BNO055 Library Documentation

Release 1.0

Radomir Dopieralski

Oct 21, 2019

Contents

CHAPTER 1

Dependencies

This driver depends on the Register and Bus Device libraries. Please ensure they are also available on the CircuitPython filesystem. This is easily achieved by downloading a library and driver bundle.

CHAPTER 2

Usage Notes

Of course, you must import the library to use it:

```
import adafruit_bno055
```

This driver takes an instantiated and active I2C object (from the `busio` or the `bitbangio` library) as an argument to its constructor. The way to create an I2C object depends on the board you are using. For boards with labeled SCL and SDA pins, you can:

```
from busio import I2C
from board import SDA, SCL

i2c = I2C(SCL, SDA)
```

Once you have the I2C object, you can create the sensor object:

```
sensor = adafruit_bno055.BNO055(i2c)
```

And then you can start reading the measurements:

```
print(sensor.temperature)
print(sensor.euler)
print(sensor.gravity)
```

CHAPTER 3

Contributing

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

CHAPTER 4

Building locally

To build this library locally you'll need to install the circuitpython-build-tools package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-bno055 --library_
↪location .
```

4.1 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the index.html in your browser to view them. It will also (due to -W) error out on any warning like Travis will. This is a good way to locally verify it will pass.

CHAPTER 5

Table of Contents

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/bno055_simpletest.py

```
import time
import board
import busio
import adafruit_bno055

i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_bno055.BNO055(i2c)

while True:
    print('Temperature: {} degrees C'.format(sensor.temperature))
    print('Accelerometer (m/s^2): {}'.format(sensor.acceleration))
    print('Magnetometer (microteslas): {}'.format(sensor.magnetic))
    print('Gyroscope (rad/sec): {}'.format(sensor.gyro))
    print('Euler angle: {}'.format(sensor.euler))
    print('Quaternion: {}'.format(sensor.quaternion))
    print('Linear acceleration (m/s^2): {}'.format(sensor.linear_acceleration))
    print('Gravity (m/s^2): {}'.format(sensor.gravity))
    print()

    time.sleep(1)
```

5.2 adafruit_bno055 - Adafruit 9-DOF Absolute Orientation IMU Fusion Breakout - BNO055

This is a CircuitPython driver for the Bosch BNO055 nine degree of freedom inertial measurement unit module with sensor fusion.

- Author(s): Radomir Dopieralski

class adafruit_bno055.**BNO055**(*i2c*, *address=40*)
Driver for the BNO055 9DOF IMU sensor.

acceleration
Gives the raw accelerometer readings, in m/s. Returns an empty tuple of length 3 when this property has been disabled by the current mode.

calibrated
Boolean indicating calibration status.

calibration_status
Tuple containing sys, gyro, accel, and mag calibration data.

euler
Gives the calculated orientation angles, in degrees. Returns an empty tuple of length 3 when this property has been disabled by the current mode.

external_crystal
Switches the use of external crystal on or off.

gravity
Returns the gravity vector, without acceleration in m/s. Returns an empty tuple of length 3 when this property has been disabled by the current mode.

gyro
Gives the raw gyroscope reading in radians per second. Returns an empty tuple of length 3 when this property has been disabled by the current mode.

linear_acceleration
Returns the linear acceleration, without gravity, in m/s. Returns an empty tuple of length 3 when this property has been disabled by the current mode.

magnetic
Gives the raw magnetometer readings in microteslas. Returns an empty tuple of length 3 when this property has been disabled by the current mode.

mode
Switch the mode of operation and return the previous mode.

Mode of operation defines which sensors are enabled and whether the measurements are absolute or relative. If a sensor is disabled, it will return an empty tuple.

legend: x=on, -=off +——————+———-+———+———+———-+ | Mode | Accel | Compass | Gyro | Absolute | +==================+=======+=========+======+==========+ | CONFIG_MODE | - | - | - | - | +——————+———-+———+———+———-+ | ACCONLY_MODE | X | - | - | - | +——————+———-+———+———+———-+ | MAGONLY_MODE | - | X | - | - | +——————+———-+———+———+———-+ | GYRONLY_MODE | - | - | X | - | +——————+———-+———+———+———-+ | ACCMAG_MODE | X | X | - | - | +——————+———-+———+———+———-+ | ACCGYRO_MODE | X | - | X | - | +——————+———-+———+———+———-+ | MAGGYRO_MODE | - | X | X | - |

+————————+———-+———+———+———-+ | AMG_MODE | X | X | X | - | +————————+———-+———+———+———-+ | IMUPLUS_MODE | X | - | X | - | +————————+———-+———+———+———-+ | COMPASS_MODE | X | X | - | X | +————————+———-+———+———+———-+ | M4G_MODE | X | X | - | - | +————————+———-+———+———+———-+ | NDOF_FMC_OFF_MODE| X | X | X | X | +————————+———-+———+———+———-+ | NDOF_MODE | X | X | X | X | +————————+———-+———+———+———-+

The default mode is `NDOF_MODE`.

quaternion
Gives the calculated orientation as a quaternion. Returns an empty tuple of length 3 when this property has been disabled by the current mode.

temperature
Measures the temperature of the chip in degrees Celsius.

use_external_crystal
Switches the use of external crystal on or off.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Python Module Index

a

Index