Adafruit VL53L0X Library Documentation

Release 1.0

Tony DiCola

Aug 03, 2018

Contents

Adafruit CircuitPython module for the LIS3DH accelerometer.

Note this module is made to work with CircuitPython and not MicroPython APIs.

CHAPTER 1

Usage Example

See the guide at: https://learn.adafruit.com/circuitpython-hardware-lis3dh-accelerometer

CHAPTER 2

Dependencies

This driver depends on:

- Adafruit CircuitPython
- Bus Device

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

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-travis-build-tools package.

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

Then run the build:

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/lis3dh_simpletest.py

```
# Accelerometer example.
# Reads the accelerometer x, y, z values and prints them every tenth of a second.
# Open the serial port after running to see the output printed.
# Author: Tony DiCola
import time
import board
import adafruit_lis3dh


# Uncomment _one_ of the hardware setups below depending on your wiring:

# Hardware I2C setup. Use the CircuitPlayground built-in accelerometer if available;
# otherwise check I2C pins.
import busio
if hasattr(board, 'ACCELEROMETER_SCL'):
    i2c = busio.I2C(board.ACCELEROMETER_SCL, board.ACCELEROMETER_SDA)
    lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, address=0x19)
else:
    i2c = busio.I2C(board.SCL, board.SDA)
    lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c)

# Software I2C setup:
#import bitbangio
#i2c = bitbangio.I2C(board.SCL, board.SDA)
#lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c)

# Hardware SPI setup:
```

(continues on next page)

(continued from previous page)

```
#import busio
#spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
#cs = busio.DigitalInOut(board.D6)  # Set to appropriate CS pin!
#lis3dh = adafruit_lis3dh.LIS3DH_SPI(spi, cs)


# Set range of accelerometer (can be RANGE_2_G, RANGE_4_G, RANGE_8_G or RANGE_16_G).
lis3dh.range = adafruit_lis3dh.RANGE_2_G

# Loop forever printing accelerometer values
while True:
    # Read accelerometer values (in m / s ^ 2).  Returns a 3-tuple of x, y,
    # z axis values.  Divide them by 9.806 to convert to Gs.
    x, y, z = [value / adafruit_lis3dh.STANDARD_GRAVITY for value in lis3dh.
↪acceleration]
    print('x = {}G, y = {}G, z = {}G'.format(x, y, z))
    # Small delay to keep things responsive but give time for interrupt processing.
    time.sleep(0.1)
```

Here are some additional examples:

Listing 2: examples/tap.py

```
# Tap detection example.
# Will loop forever printing when a single or double click is detected.
# Open the serial port after running to see the output printed.
# Author: Tony DiCola
import board
import adafruit_lis3dh
import busio
import digitalio


# Uncomment _one_ of the hardware setups below depending on your wiring:

# Hardware I2C setup. Use the CircuitPlayground built-in accelerometer if available;
# otherwise check I2C pins.
if hasattr(board, 'ACCELEROMETER_SCL'):
    i2c = busio.I2C(board.ACCELEROMETER_SCL, board.ACCELEROMETER_SDA)
    int1 = digitalio.DigitalInOut(board.ACCELEROMETER_INTERRUPT)
    lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, address=0x19, int1=int1)
else:
    i2c = busio.I2C(board.SCL, board.SDA)
    int1 = digitalio.DigitalInOut(board.D10)  # Set this to the correct pin for the␣
↪interrupt!
    lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, int1=int1)

# Software I2C setup:
# import bitbangio
# i2c = bitbangio.I2C(board.SCL, board.SDA)
# lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c)

# Hardware SPI setup:
# import busio
# spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
# cs = busio.DigitalInOut(board.D6)  # Set to appropriate CS pin!
# lis3dh = adafruit_lis3dh.LIS3DH_SPI(spi, cs)
```

(continues on next page)

(continued from previous page)

```

# Set range of accelerometer (can be RANGE_2_G, RANGE_4_G, RANGE_8_G or RANGE_16_G).
lis3dh.range = adafruit_lis3dh.RANGE_8_G

# Set tap detection to double taps.  The first parameter is a value:
#  - 0 = Disable tap detection.
#  - 1 = Detect single taps.
#  - 2 = Detect double taps.
# The second parameter is the threshold and a higher value means less sensitive
# tap detection.  Note the threshold should be set based on the range above:
#  - 2G = 40-80 threshold
#  - 4G = 20-40 threshold
#  - 8G = 10-20 threshold
#  - 16G = 5-10 threshold
lis3dh.set_tap(2, 60)

# Loop forever printing if a double tap is detected.
while True:
    if lis3dh.tapped:
        print('Tapped!')
```

Listing 3: examples/adc.py

```
# Analog to digital converter example.
# Will loop forever printing ADC channel 1 raw and mV values every second.
# Open the serial port after running to see the output printed.
# NOTE the ADC can only read voltages in the range of ~900mV to 1200mV!
# Author: Tony DiCola
import time
import board
import adafruit_lis3dh


# Uncomment _one_ of the hardware setups below depending on your wiring:

# Hardware I2C setup:
import busio
i2c = busio.I2C(board.SCL, board.SDA)
lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c)

# Software I2C setup:
#import bitbangio
#i2c = bitbangio.I2C(board.SCL, board.SDA)
#lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c)

# Hardware SPI setup:
#import busio
#spi = busio.SPI(board.SCK, board.MOSI, board.MISO)
#cs = busio.DigitalInOut(board.D6)  # Set to appropriate CS pin!
#lis3dh = adafruit_lis3dh.LIS3DH_SPI(spi, cs)


# Loop forever printing ADC readings.
while True:
    # Read raw ADC value.  Specify which ADC to read: 1, 2, or 3.
    adc1_raw = lis3dh.read_adc_raw(1)
```

(continues on next page)

(continued from previous page)

```
    # Or read the ADC value in millivolts:
    adc1_mV = lis3dh.read_adc_mV(1)
    print('ADC 1 = {} ({} mV)'.format(adc1_raw, adc1_mV))
    time.sleep(1)
```

Listing 4: examples/spinner.py

```
# Circuit Playground Express CircuitPython Fidget Spinner
# This is meant to work with the Circuit Playground Express board:
#   https://www.adafruit.com/product/3333
# Needs this LIS3DH module and the NeoPixel module installed:
#   https://github.com/adafruit/Adafruit_CircuitPython_LIS3DH
#   https://github.com/adafruit/Adafruit_CircuitPython_NeoPixel
# Author: Tony DiCola
# License: MIT License (https://opensource.org/licenses/MIT)
# pylint: disable=redefined-outer-name
import math
import time

import board
import busio

import adafruit_lis3dh
import neopixel

from micropython import const

# Configuration:
ACCEL_RANGE = adafruit_lis3dh.RANGE_16_G  # Accelerometer range.
TAP_THRESHOLD = 20              # Accelerometer tap threshold.  Higher values
                                # mean you need to tap harder to start a spin.
SPINNER_DECAY = 0.5             # Decay rate for the spinner.  Set to a value
                                # from 0 to 1.0 where lower values mean the
                                # spinner slows down faster.
PRIMARY_COLOR = (0, 255, 0)     # Color of the spinner dots.
SECONDARY_COLOR = (0, 0, 0)     # Background color of the spinner.


# Define a class that represents the fidget spinner.
class FidgetSpinner:

    def __init__(self, decay=0.5):
        self._decay = decay
        self._velocity = 0.0
        self._elapsed = 0.0
        self._position = 0.0

    def spin(self, velocity):
        self._velocity = velocity
        self._elapsed = 0.0

    def get_position(self, delta):
        # Increment elapsed time and compute the current velocity after a
        # decay of the initial velocity.
        self._elapsed += delta
        current_velocity = self._velocity*math.pow(self._decay, self._elapsed)
```

(continues on next page)

(continued from previous page)

```
        self._position += current_velocity*delta
        # Make sure the position stays within values that range from 0 to <10.
        self._position = math.fmod(self._position, 10.0)
        if self._position < 0.0:
            self._position += 10.0
        return self._position


# Initialize NeoPixels and accelerometer.
pixels = neopixel.NeoPixel(board.NEOPIXEL, 10, auto_write=False)
pixels.fill((0, 0, 0))
pixels.show()
i2c = busio.I2C(board.ACCELEROMETER_SCL, board.ACCELEROMETER_SDA)
lis3dh = adafruit_lis3dh.LIS3DH_I2C(i2c, address=25)

# Set accelerometer range.
lis3dh.range = ACCEL_RANGE
# Enable single click detection, but use a custom CLICK_CFG register value
# to only detect clicks on the X axis (instead of all 3 X, Y, Z axes).
lis3dh.set_tap(1, TAP_THRESHOLD, click_cfg=0x01)
# Enable LIS3DH FIFO in stream mode.  This reaches in to the LIS3DH library to
# call internal methods that change a few register values.  This must be done
# AFTER calling set_tap above because the set_tap function also changes
# REG_CTRL5.
# Define register numbers, which are not exported from the library.
_REG_CTRL5 = const(0x24)
_REG_CLICKSRC = const(0x39)
# pylint: disable=protected-access
lis3dh._write_register_byte(_REG_CTRL5, 0b01001000)
lis3dh._write_register_byte(0x2E, 0b10000000)  # Set FIFO_CTRL to Stream mode.
# pylint: disable=protected-access

# Create a fidget spinner object.
spinner = FidgetSpinner(SPINNER_DECAY)


# Main loop will run forever checking for click/taps from accelerometer and
# then spinning the spinner.
last = time.monotonic()  # Keep track of the last time the loop ran.
while True:
    # Read the raw click detection register value and check if there was
    # a click detected.
    clicksrc = lis3dh._read_register_byte(_REG_CLICKSRC) # pylint: disable=protected-
↪access
    if clicksrc & 0b01000000 > 0:
        # Click was detected!  Quickly read 32 values from the accelerometer
        # FIFO and look for the maximum magnitude values.
        maxval = lis3dh.acceleration[0]  # Grab just the X acceleration value.
        for i in range(31):
            x = abs(lis3dh.acceleration[0])
            if x > maxval:
                maxval = x
        # Check if this was a positive or negative spin/click event.
        if clicksrc == 0b1010001:
            # Positive click, spin in a positive direction.
            spinner.spin(maxval)
        elif clicksrc == 0b1011001:
```

(continues on next page)

(continued from previous page)

```
            # Negative click, spin in negative direction.
            spinner.spin(-maxval)
    # Update the amount of time that's passed since the last loop iteration.
    current = time.monotonic()
    delta = current - last
    last = current
    # Set all pixels to secondary color.
    pixels.fill(SECONDARY_COLOR)
    # Update the fidget spinner position and turn on the appropriate pixels.
    pos = int(spinner.get_position(delta))
    # Set the current position pixel and the pixel exactly opposite it (i.e. 5
    # pixels ahead, wrapping back to the start) to the primary color.
    pixels[pos] = PRIMARY_COLOR
    pixels[(pos + 5) % 10] = PRIMARY_COLOR
    pixels.show()
    # Small delay to stay responsive but give time for interrupt processing.
    time.sleep(0.05)
```

CHAPTER 6

Indices and tables

- genindex
- modindex
- search