
Adafruit
CIRCUITPYTHON *ADS1X15 Library Documentation*
Release 1.0

Carter Nelson

Jun 07, 2021

Contents

1	Installation & Dependencies	3
1.1	Installing from PyPI	3
2	Usage Example	5
2.1	Single Ended	5
3	Contributing	7
4	Documentation	9
5	Table of Contents	11
5.1	Simple test	11
5.2	ads1x15	12
5.3	ads1015	13
5.4	ads1115	13
5.5	analog_in	14
6	Indices and tables	15
	Python Module Index	17
	Index	19

Support for the ADS1x15 series of analog-to-digital converters. Available in 12-bit (ADS1015) and 16-bit (ADS1115) versions.

Installation & Dependencies

This driver depends on:

- [Adafruit CircuitPython](#)
- [Bus Device](#)

Please ensure all dependencies are available on the CircuitPython filesystem. This can be most easily achieved by downloading and installing [the Adafruit library and driver bundle](#) on your device.

1.1 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-ads1x15
```

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

```
sudo pip3 install adafruit-circuitpython-ads1x15
```

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-ads1x15
```


2.1 Single Ended

```
import time
import board
import busio
import adafruit_ads1x15.ads1015 as ADS
from adafruit_ads1x15.analog_in import AnalogIn

# Create the I2C bus
i2c = busio.I2C(board.SCL, board.SDA)

# Create the ADC object using the I2C bus
ads = ADS.ADS1015(i2c)

# Create single-ended input on channel 0
chan = AnalogIn(ads, ADS.P0)

# Create differential input between channel 0 and 1
#chan = AnalogIn(ads, ADS.P0, ADS.P1)

print("{:>5}\t{:>5}".format('raw', 'v'))

while True:
    print("{:>5}\t{:>5.3f}".format(chan.value, chan.voltage))
    time.sleep(0.5)
```


CHAPTER 3

Contributing

Contributions are welcome! Please read our [Code of Conduct](#) before contributing to help this project stay welcoming.

CHAPTER 4

Documentation

For information on building library documentation, please check out [this guide](#).

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/ads1x15_ads1015_simpletest.py

```
1  # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2  # SPDX-License-Identifier: MIT
3
4  import time
5  import board
6  import busio
7  import adafruit_ads1x15.ads1015 as ADS
8  from adafruit_ads1x15.analog_in import AnalogIn
9
10 # Create the I2C bus
11 i2c = busio.I2C(board.SCL, board.SDA)
12
13 # Create the ADC object using the I2C bus
14 ads = ADS.ADS1015(i2c)
15
16 # Create single-ended input on channel 0
17 chan = AnalogIn(ads, ADS.P0)
18
19 # Create differential input between channel 0 and 1
20 # chan = AnalogIn(ads, ADS.P0, ADS.P1)
21
22 print("{:>5}\t{:>5}".format("raw", "v"))
23
24 while True:
25     print("{:>5}\t{:>5.3f}".format(chan.value, chan.voltage))
26     time.sleep(0.5)
```

Listing 2: examples/ads1x15_ads1115_simpletest.py

```

1 # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2 # SPDX-License-Identifier: MIT
3
4 import time
5 import board
6 import busio
7 import adafruit_ads1x15.ads1115 as ADS
8 from adafruit_ads1x15.analog_in import AnalogIn
9
10 # Create the I2C bus
11 i2c = busio.I2C(board.SCL, board.SDA)
12
13 # Create the ADC object using the I2C bus
14 ads = ADS.ADS1115(i2c)
15 # you can specify an I2C address instead of the default 0x48
16 # ads = ADS.ADS1115(i2c, address=0x49)
17
18 # Create single-ended input on channel 0
19 chan = AnalogIn(ads, ADS.P0)
20
21 # Create differential input between channel 0 and 1
22 # chan = AnalogIn(ads, ADS.P0, ADS.P1)
23
24 print("{:>5}\t{:>5}".format("raw", "v"))
25
26 while True:
27     print("{:>5}\t{:>5.3f}".format(chan.value, chan.voltage))
28     time.sleep(0.5)

```

5.2 ads1x15

CircuitPython base class driver for ADS1015/1115 ADCs.

- Author(s): Carter Nelson

class adafruit_ads1x15.ads1x15.**ADS1x15** (*i2c*, *gain=1*, *data_rate=None*, *mode=256*, *address=72*)

Base functionality for ADS1x15 analog to digital converters.

data_rate

The data rate for ADC conversion in samples per second.

gain

The ADC gain.

gains

Possible gain settings.

get_last_result (*fast=False*)

Read the last conversion result when in continuous conversion mode. Will return a signed integer value. If *fast* is True, the register pointer is not updated as part of the read. This reduces I2C traffic and increases possible read rate.

mode

The ADC conversion mode.

rate_config

Rate configuration masks.

rates

Possible data rate settings.

read (*pin, is_differential=False*)

I2C Interface for ADS1x15-based ADCs reads.

params:

param pin individual or differential pin.

param bool is_differential single-ended or differential read.

class adafruit_ads1x15.ads1x15.**Mode**

An enum-like class representing possible ADC operating modes.

5.3 ads1015

CircuitPython driver for ADS1015 ADCs.

- Author(s): Carter Nelson

class adafruit_ads1x15.ads1015.**ADS1015** (*i2c, gain=1, data_rate=None, mode=256, address=72*)

Class for the ADS1015 12 bit ADC.

bits

The ADC bit resolution.

rate_config

Rate configuration masks.

rates

Possible data rate settings.

5.4 ads1115

CircuitPython driver for 1115 ADCs.

- Author(s): Carter Nelson

class adafruit_ads1x15.ads1115.**ADS1115** (*i2c, gain=1, data_rate=None, mode=256, address=72*)

Class for the ADS1115 16 bit ADC.

bits

The ADC bit resolution.

rate_config

Rate configuration masks.

rates

Possible data rate settings.

5.5 analog_in

AnalogIn for single-ended and differential ADC readings.

- Author(s): Carter Nelson, adapted from MCP3xxx original by Brent Rubell

class adafruit_ads1x15.analog_in.**AnalogIn** (*ads, positive_pin, negative_pin=None*)
AnalogIn Mock Implementation for ADC Reads.

value

Returns the value of an ADC pin as an integer.

voltage

Returns the voltage from the ADC pin as a floating point value.

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`

a

adafruit_ads1x15.ads1015, 13
adafruit_ads1x15.ads1115, 13
adafruit_ads1x15.ads1x15, 12
adafruit_ads1x15.analog_in, 13

A

adafruit_ads1x15.ads1015 (module), 13
adafruit_ads1x15.ads1115 (module), 13
adafruit_ads1x15.ads1x15 (module), 12
adafruit_ads1x15.analog_in (module), 13
ADS1015 (class in adafruit_ads1x15.ads1015), 13
ADS1115 (class in adafruit_ads1x15.ads1115), 13
ADS1x15 (class in adafruit_ads1x15.ads1x15), 12
AnalogIn (class in adafruit_ads1x15.analog_in), 14

B

bits (adafruit_ads1x15.ads1015.ADS1015 attribute), 13
bits (adafruit_ads1x15.ads1115.ADS1115 attribute), 13

D

data_rate (adafruit_ads1x15.ads1x15.ADS1x15 attribute), 12

G

gain (adafruit_ads1x15.ads1x15.ADS1x15 attribute), 12
gains (adafruit_ads1x15.ads1x15.ADS1x15 attribute), 12
get_last_result ()
(adafruit_ads1x15.ads1x15.ADS1x15 method), 12

M

mode (adafruit_ads1x15.ads1x15.ADS1x15 attribute), 12
Mode (class in adafruit_ads1x15.ads1x15), 13

R

rate_config (adafruit_ads1x15.ads1015.ADS1015 attribute), 13
rate_config (adafruit_ads1x15.ads1115.ADS1115 attribute), 13

rate_config (adafruit_ads1x15.ads1x15.ADS1x15 attribute), 12
rates (adafruit_ads1x15.ads1015.ADS1015 attribute), 13
rates (adafruit_ads1x15.ads1115.ADS1115 attribute), 13
rates (adafruit_ads1x15.ads1x15.ADS1x15 attribute), 13
read () (adafruit_ads1x15.ads1x15.ADS1x15 method), 13

V

value (adafruit_ads1x15.analog_in.AnalogIn attribute), 14
voltage (adafruit_ads1x15.analog_in.AnalogIn attribute), 14