
AdafruitOneWire Library Documentation

Release 1.0

Carter Nelson

Dec 21, 2018

Contents

1	Dependencies	3
2	Usage Example	5
3	Contributing	7
4	Building locally	9
4.1	Sphinx documentation	9
5	Table of Contents	11
5.1	Simple test	11
5.2	adafruit_owewire.bus	12
5.3	adafruit_owewire.device	13
6	Indices and tables	15
	Python Module Index	17

Classes for use in communicating with devices on a 1-Wire bus.

CHAPTER 1

Dependencies

This driver depends on:

- [Adafruit CircuitPython](#)

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

CHAPTER 2

Usage Example

```
import board
from adafruit_owewire.bus import OneWireBus
ow_bus = OneWireBus(board.D2)
devices = ow_bus.scan()
for d in devices:
    print("ROM={}\tFamily=0x{:02x}".format(d.rom, d.family_code))
```


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-onewire --
↳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.

5.1 Simple test

Ensure your device works with this simple test.

Listing 1: examples/onewire_demo.py

```
1 import board
2 from adafruit_onewire.bus import OneWireBus
3
4 # Create the 1-Wire Bus
5 # Use whatever pin you've connected to on your board
6 ow_bus = OneWireBus(board.D2)
7
8 # Reset and check for presence pulse.
9 # This is basically - "is there anything out there?"
10 print("Resetting bus...", end='')
11 if ow_bus.reset():
12     print("OK.")
13 else:
14     raise RuntimeError("Nothing found on bus.")
15
16 # Run a scan to get all of the device ROM values
17 print("Scanning for devices...", end='')
18 devices = ow_bus.scan()
19 print("OK.")
20 print("Found {} device(s)".format(len(devices)))
21
22 # For each device found, print out some info
23 for i, d in enumerate(devices):
24     print("Device {:>3}".format(i))
25     print("\tSerial Number = ", end='')
26     for byte in d.serial_number:
27         print("0x{:02x} ".format(byte), end='')
```

(continues on next page)

(continued from previous page)

```
28     print("\n\tFamily = 0x{:02x}".format(d.family_code))
29
30 # Usage beyond this is device specific. See a CircuitPython library for a 1-Wire
31 # device for examples and how OneWireDevice is used.
```

5.2 adafruit_owewire.bus

Provide access to a 1-Wire bus.

- Author(s): Carter Nelson

class adafruit_owewire.bus.OneWireAddress(*rom*)

A class to represent a 1-Wire address.

crc

The 8 bit CRC.

family_code

The 8 bit family code.

rom

The unique 64 bit ROM code.

serial_number

The 48 bit serial number.

class adafruit_owewire.bus.OneWireBus(*pin*)

A class to represent a 1-Wire bus.

static crc8(*data*)

Perform the 1-Wire CRC check on the provided data.

Parameters *data* (*bytearray*) – 8 byte array representing 64 bit ROM code

maximum_devices

The maximum number of devices the bus will scan for. Valid range is 1 to 255. It is an error to have more devices on the bus than this number. Having less is OK.

readinto(*buf*, *, *start=0*, *end=None*)

Read into *buf* from the device. The number of bytes read will be the length of *buf*.

If *start* or *end* is provided, then the buffer will be sliced as if *buf[start:end]*. This will not cause an allocation like *buf[start:end]* will so it saves memory.

Parameters

- **buf** (*bytearray*) – buffer to write into
- **start** (*int*) – Index to start writing at
- **end** (*int*) – Index to write up to but not include

reset(*required=False*)

Perform a reset and check for presence pulse.

Parameters **required** (*bool*) – require presence pulse

scan()

Scan for devices on the bus and return a list of addresses.

write (*buf*, *, *start=0*, *end=None*)

Write the bytes from *buf* to the device.

If *start* or *end* is provided, then the buffer will be sliced as if `buffer[start:end]`. This will not cause an allocation like `buffer[start:end]` will so it saves memory.

Parameters

- **buf** (*bytearray*) – buffer containing the bytes to write
- **start** (*int*) – Index to start writing from
- **end** (*int*) – Index to read up to but not include

exception `adafruit_onewire.bus.OneWireError`

A class to represent a 1-Wire exception.

5.3 `adafruit_onewire.device`

Provides access to a single device on the 1-Wire bus.

- Author(s): Carter Nelson

class `adafruit_onewire.device.OneWireDevice` (*bus*, *address*)

A class to represent a single device on the 1-Wire bus.

readinto (*buf*, *, *start=0*, *end=None*)

Read into *buf* from the device. The number of bytes read will be the length of *buf*.

If *start* or *end* is provided, then the buffer will be sliced as if `buf[start:end]`. This will not cause an allocation like `buf[start:end]` will so it saves memory.

Parameters

- **buf** (*bytearray*) – buffer to write into
- **start** (*int*) – Index to start writing at
- **end** (*int*) – Index to write up to but not include

write (*buf*, *, *start=0*, *end=None*)

Write the bytes from *buf* to the device.

If *start* or *end* is provided, then the buffer will be sliced as if `buffer[start:end]`. This will not cause an allocation like `buffer[start:end]` will so it saves memory.

Parameters

- **buf** (*bytearray*) – buffer containing the bytes to write
- **start** (*int*) – Index to start writing from
- **end** (*int*) – Index to read up to but not include

CHAPTER 6

Indices and tables

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

a

`adafruit_owewire.bus`, [12](#)

`adafruit_owewire.device`, [13](#)

A

adafruit_owewire.bus (module), [12](#)
adafruit_owewire.device (module), [13](#)

C

crc (adafruit_owewire.bus.OneWireAddress attribute), [12](#)
crc8() (adafruit_owewire.bus.OneWireBus static method),
[12](#)

F

family_code (adafruit_owewire.bus.OneWireAddress attribute), [12](#)

M

maximum_devices (adafruit_owewire.bus.OneWireBus attribute), [12](#)

O

OneWireAddress (class in adafruit_owewire.bus), [12](#)
OneWireBus (class in adafruit_owewire.bus), [12](#)
OneWireDevice (class in adafruit_owewire.device), [13](#)
OneWireError, [13](#)

R

readinto() (adafruit_owewire.bus.OneWireBus method),
[12](#)
readinto() (adafruit_owewire.device.OneWireDevice method), [13](#)
reset() (adafruit_owewire.bus.OneWireBus method), [12](#)
rom (adafruit_owewire.bus.OneWireAddress attribute),
[12](#)

S

scan() (adafruit_owewire.bus.OneWireBus method), [12](#)
serial_number (adafruit_owewire.bus.OneWireAddress attribute), [12](#)

W

write() (adafruit_owewire.bus.OneWireBus method), [12](#)

write() (adafruit_owewire.device.OneWireDevice method), [13](#)