

---

# **AdafruitOneWire Library Documentation**

***Release 1.0***

**Carter Nelson**

**Feb 10, 2021**



---

## Contents

---

<b>1</b>	<b>Dependencies</b>	<b>3</b>
<b>2</b>	<b>Usage Example</b>	<b>5</b>
<b>3</b>	<b>Contributing</b>	<b>7</b>
<b>4</b>	<b>Documentation</b>	<b>9</b>
<b>5</b>	<b>Table of Contents</b>	<b>11</b>
5.1	Simple test . . . . .	11
5.2	adafruit_owewire.bus . . . . .	12
5.3	adafruit_owewire.device . . . . .	13
<b>6</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>
	<b>Index</b>	<b>19</b>



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

---

### 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/onewire\_simpletest.py

```
1  # SPDX-FileCopyrightText: 2021 ladyada for Adafruit Industries
2  # SPDX-License-Identifier: MIT
3
4  import board
5  from adafruit_onewire.bus import OneWireBus
6
7  # Create the 1-Wire Bus
8  # Use whatever pin you've connected to on your board
9  ow_bus = OneWireBus(board.D2)
10
11 # Reset and check for presence pulse.
12 # This is basically - "is there anything out there?"
13 print("Resetting bus...", end="")
14 if ow_bus.reset():
15     print("OK.")
16 else:
17     raise RuntimeError("Nothing found on bus.")
18
19 # Run a scan to get all of the device ROM values
20 print("Scanning for devices...", end="")
21 devices = ow_bus.scan()
22 print("OK.")
23 print("Found {} device(s)".format(len(devices)))
24
25 # For each device found, print out some info
26 for i, d in enumerate(devices):
27     print("Device {:>3}".format(i))
```

(continues on next page)

(continued from previous page)

```

28     print("\tSerial Number = ", end="")
29     for byte in d.serial_number:
30         print("0x{:02x} ".format(byte), end="")
31     print("\n\tFamily = 0x{:02x}".format(d.family_code))
32
33     # Usage beyond this is device specific. See a CircuitPython library for a 1-Wire
34     # 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),  
[13](#)  
`write()` (`adafruit_owewire.device.OneWireDevice`  
method), [13](#)