

---

# **Adafruit IRREMOTE Library Documentation**

***Release 1.0***

**Scott Shawcroft**

**Dec 21, 2018**



---

## 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>Building locally</b>	<b>9</b>
4.1	Sphinx documentation . . . . .	9
<b>5</b>	<b>Table of Contents</b>	<b>11</b>
5.1	Simple test . . . . .	11
5.2	adafruit_irremote . . . . .	12
5.2.1	Implementation Notes . . . . .	12
<b>6</b>	<b>Indices and tables</b>	<b>15</b>
	<b>Python Module Index</b>	<b>17</b>



CircuitPython driver for use with IR Receivers.

Examples of products to use this library with:

- [Circuit Playground Express](#)
- [IR Receiver Sensor](#)



# 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

---

```
# Circuit Playground Express Demo Code
# Adjust the pulseio 'board.PIN' if using something else
import pulseio
import board
import adafruit_irremote
pulsein = pulseio.PulseIn(board.REMOTEIN, maxlen=120, idle_state=True)
decoder = adafruit_irremote.GenericDecode()

# size must match what you are decoding! for NEC use 4
received_code = bytearray(4)

while True:
    pulses = decoder.read_pulses(pulsein)
    print("Heard", len(pulses), "Pulses:", pulses)
    try:
        code = decoder.decode_bits(pulses, debug=False)
        print("Decoded:", code)
    except adafruit_irremote.IRNECRepeatException: # unusual short code!
        print("NEC repeat!")
    except adafruit_irremote.IRDecodeException as e: # failed to decode
        print("Failed to decode: ", e.args)

    print("-----")
```



## 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-irremote --
↳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/irremote\_simpletest.py

```
1  # Circuit Playground Express Demo Code
2  # Adjust the pulseio 'board.PIN' if using something else
3  import pulseio
4  import board
5  import adafruit_irremote
6  pulsein = pulseio.PulseIn(board.REMOTEIN, maxlen=120, idle_state=True)
7  decoder = adafruit_irremote.GenericDecode()
8
9  # size must match what you are decoding! for NEC use 4
10 received_code = bytearray(4)
11
12 while True:
13     pulses = decoder.read_pulses(pulsein)
14     print("Heard", len(pulses), "Pulses:", pulses)
15     try:
16         code = decoder.decode_bits(pulses, debug=False)
17         print("Decoded:", code)
18     except adafruit_irremote.IRNECRepeatException: # unusual short code!
19         print("NEC repeat!")
20     except adafruit_irremote.IRDecodeException as e: # failed to decode
21         print("Failed to decode: ", e.args)
22
23     print("-----")
```

## 5.2 adafruit\_irremote

Demo code for Circuit Playground Express:

```
# Circuit Playground Express Demo Code
# Adjust the pulseio 'board.PIN' if using something else
import pulseio
import board
import adafruit_irremote
pulsein = pulseio.PulseIn(board.REMOTEIN, maxlen=120, idle_state=True)
decoder = adafruit_irremote.GenericDecode()

# size must match what you are decoding! for NEC use 4
received_code = bytearray(4)

while True:
    pulses = decoder.read_pulses(pulsein)
    print("Heard", len(pulses), "Pulses:", pulses)
    try:
        code = decoder.decode_bits(pulses, debug=False)
        print("Decoded:", code)
    except adafruit_irremote.IRNECRepeatException: # unusual short code!
        print("NEC repeat!")
    except adafruit_irremote.IRDecodeException as e: # failed to decode
        print("Failed to decode: ", e.args)

    print("-----")
```

- Author(s): Scott Shawcroft

### 5.2.1 Implementation Notes

#### Hardware:

- Circuit Playground Express
- IR Receiver Sensor

#### Software and Dependencies:

- Adafruit CircuitPython firmware for the ESP8622 and M0-based boards: <https://github.com/adafruit/circuitpython/releases>

#### **class** adafruit\_irremote.GenericDecode

Generic decoding of infrared signals

##### **bin\_data** (*pulses*)

Compute bins of pulse lengths where pulses are  $\pm 25\%$  of the average.

**Parameters** *pulses* (*list*) – Input pulse lengths

##### **decode\_bits** (*pulses*, *debug=False*)

Decode the pulses into bits.

##### **read\_pulses** (*input\_pulses*, *max\_pulse=10000*, *blocking=True*)

Read out a burst of pulses until a pulse is longer than *max\_pulse*.

##### **Parameters**

- **input\_pulses** (*PulseIn*) – Object to read pulses from



- **max\_pulse** (*int*) – Pulse duration to end a burst
- **blocking** (*bool*) – If True, will block until pulses found. If False, will return None if no pulses. Defaults to True for backwards compatibility

**class** adafruit\_irremote.**GenericTransmit** (*header, one, zero, trail*)  
 Generic infrared transmit class that handles encoding.

**transmit** (*pulseout, data*)  
 Transmit the data using the pulseout.

#### Parameters

- **pulseout** (*pulseio.PulseOut*) – PulseOut to transmit on
- **data** (*bytearray*) – Data to transmit

**exception** adafruit\_irremote.**IRDecodeException**  
 Generic decode exception

**exception** adafruit\_irremote.**IRNECRepeatException**  
 Exception when a NEC repeat is decoded



## CHAPTER 6

---

### Indices and tables

---

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



### a

adafruit\_irremote, [11](#)



### A

`adafruit_irremote` (module), [11](#)

### B

`bin_data()` (`adafruit_irremote.GenericDecode` method),  
[12](#)

### D

`decode_bits()` (`adafruit_irremote.GenericDecode`  
method), [12](#)

### G

`GenericDecode` (class in `adafruit_irremote`), [12](#)

`GenericTransmit` (class in `adafruit_irremote`), [13](#)

### I

`IRDecodeException`, [13](#)

`IRNECRepeatException`, [13](#)

### R

`read_pulses()` (`adafruit_irremote.GenericDecode`  
method), [12](#)

### T

`transmit()` (`adafruit_irremote.GenericTransmit` method),  
[13](#)