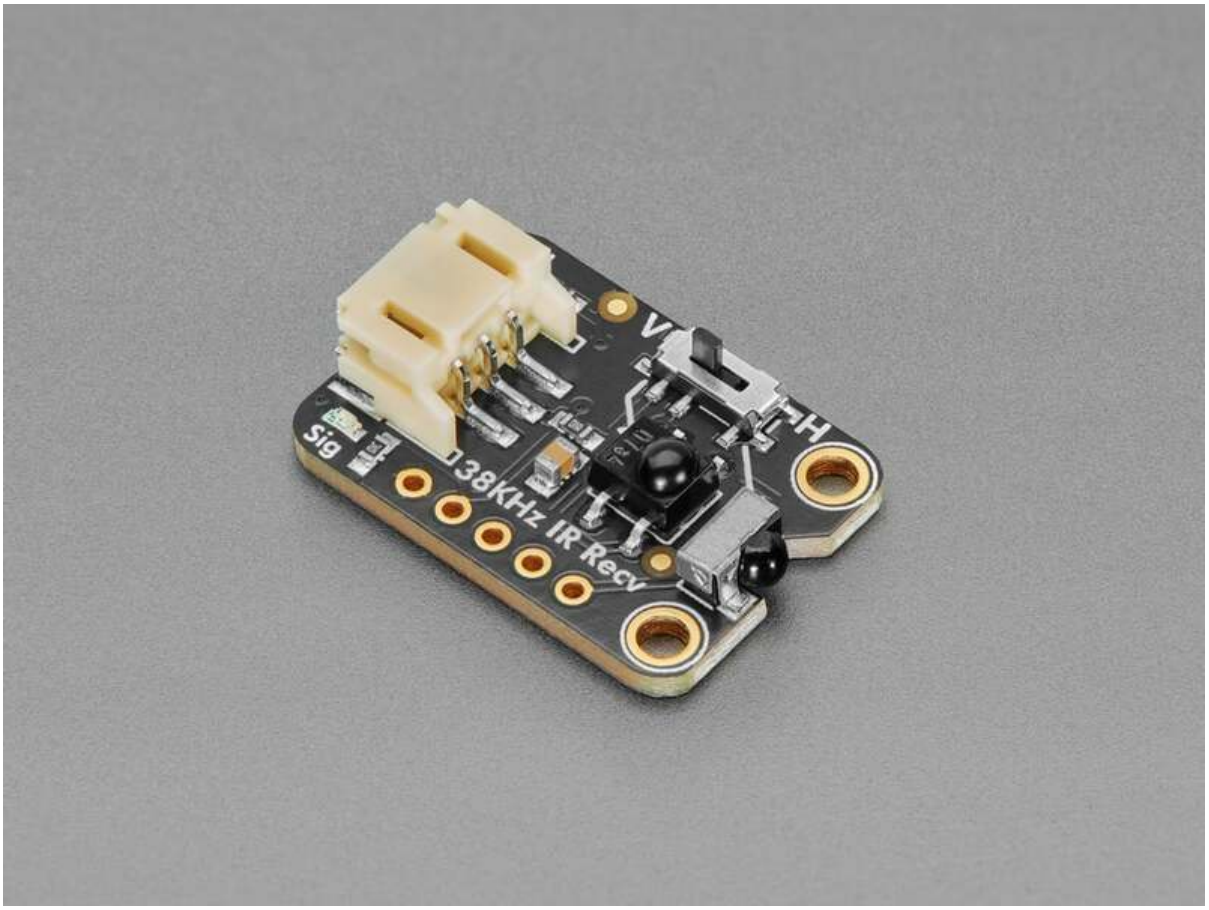
Adafruit Infrared IR Remote Receiver

Created by Liz Clark

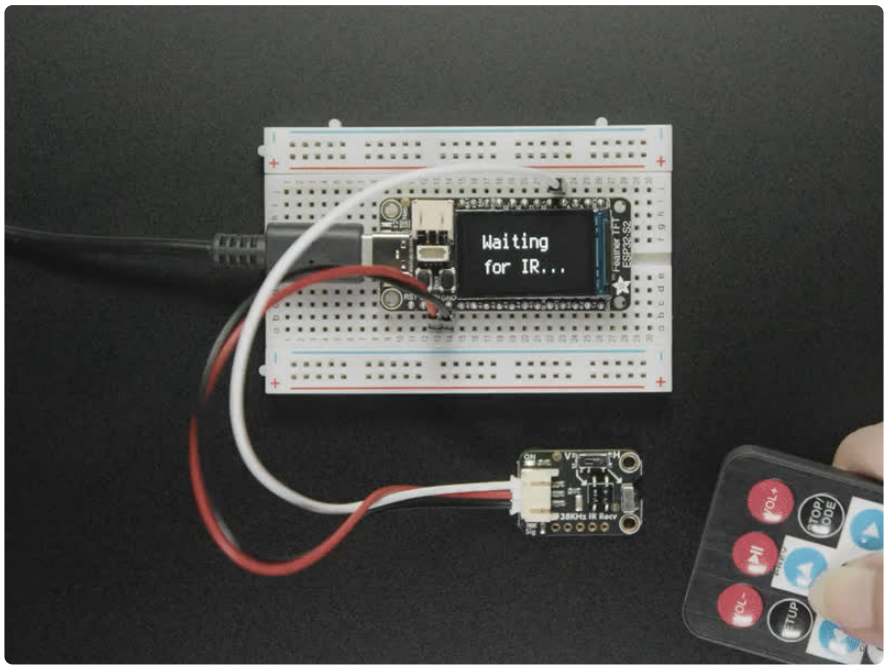


https://learn.adafruit.com/adafruit-infrared-ir-remote-receiver

Last updated on 2024-06-10 09:20:09 AM EDT

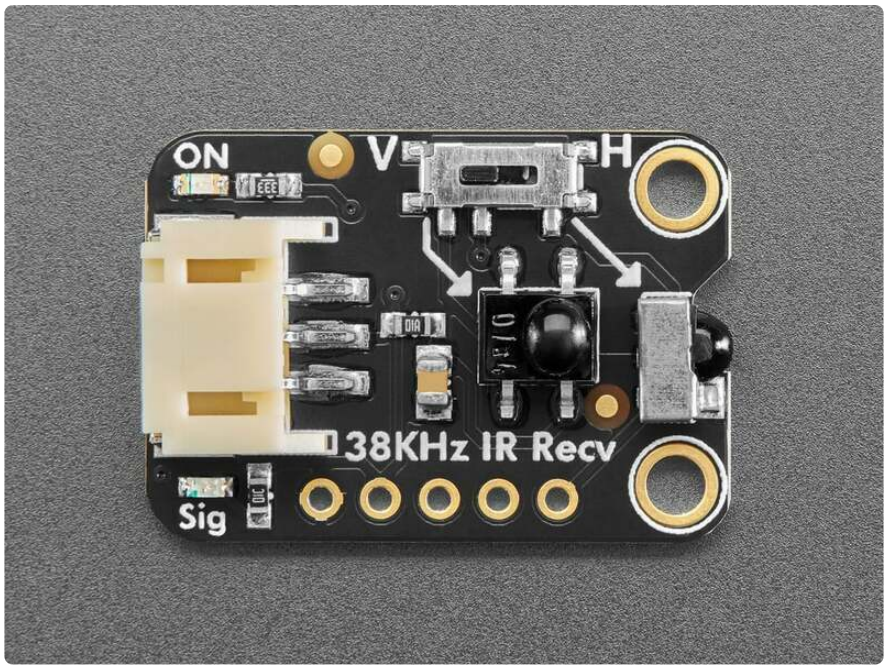
Table of Contents

Overview



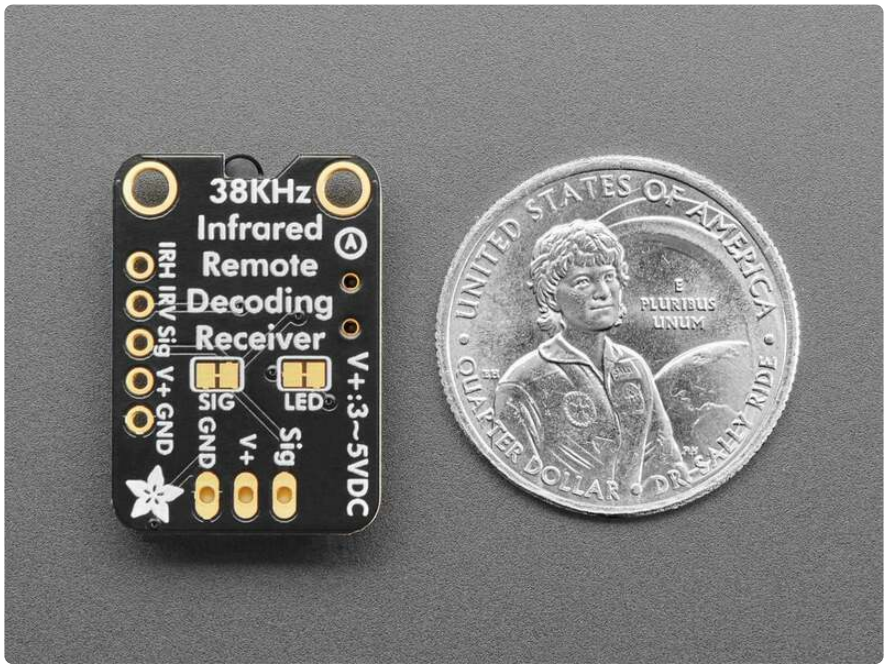
A year ago we designed a high-current-output Infrared Transmitter STEMMA (http://adafru.it/5639) which makes it easy to create high-powered IR LED blasters. Now we've sat down to design the other side, the super sensitive wide-range **Adafruit Infrared IR Remote Receiver** with two selectable IR receiver chips.

This board is specifically for receiving 38KHz IR remote control signals - it isn't going to work for proximity/distance sensing or other frequency signals



We found plenty of 38KHz receiver sensors that would work nicely on this breakout board - but when it came to choosing one that was vertical or horizontal we just couldn't make up our mind...so why not both? We've placed one on the end and one in the middle, and a slide switch to select which one you want to read the signal from.

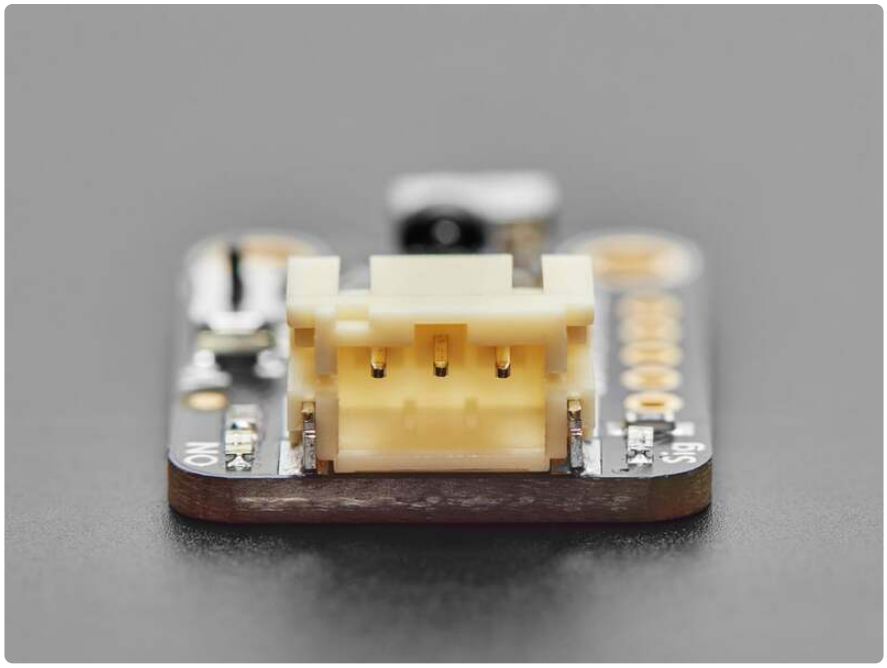
We can't just tie the outputs together because they'd 'fight' each other and give incoherent output, but if you're willing to solder two wires, it's possible to read each one independently thanks to labelled breakout pads.



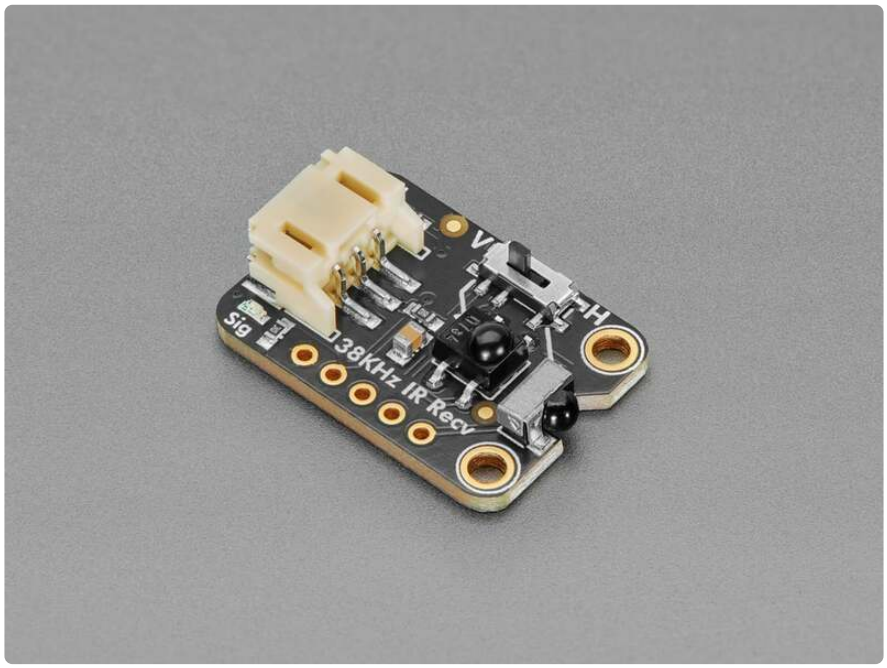
Usage is simple: Power the board by connecting V+ and ground to 3 to 5VDC, point a 38KHz remote control at the sensors and press some buttons. The demodulated IR

envelope is piped out the Signal pin into your microcontroller which will then need to decode it.

To make usage really easy, we have a green 'power good' LED and a red 'signal' LED. When IR remote signals are read by the onboard sensors, the red LED will blink to let you know.



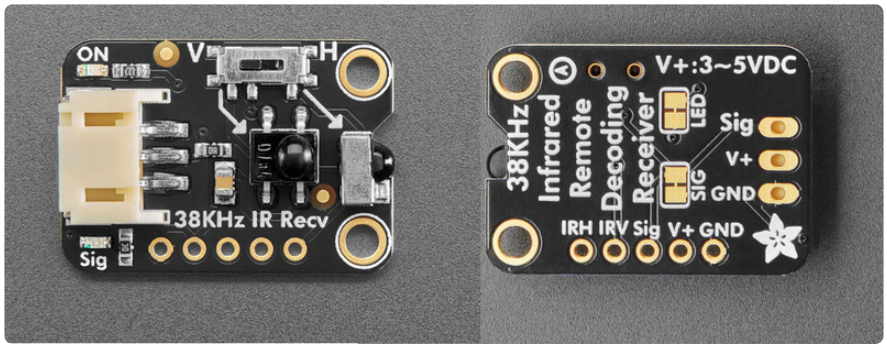
This board will work nicely for a variety of IR remote receiving projects, and with mounting holes and a cable, a lot easier to mount in enclosures and on devices. Using a 2mm pitch STEMMA JST PH cable with headers or alligator clips on the end, you can easily wire this board without any soldering.



Note that this board is specifically for receiving 38KHz IR remote control signals - it isn't going to work for proximity/distance sensing or other frequency signals. The signal must be read by a microcontroller that has pulse-input reading capabilities - basically just check that it supports common IR Receiver connectivity and decoding. Sometimes you may need to use special code or microcontroller pins.

Each STEMMA board is a fully assembled and tested PCB but no cable. No soldering is required to use it, but you will need to pick up a 2mm pitch, 3-pin STEMMA JST PH cable (https://adafru.it/18cS). Alternatively, if you do want to solder, there's a 0.1" spaced header for power/ground/signal.

Pinouts



Power Pins

- **V+** - this is the power pin. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V microcontroller like Arduino, use 5V.
- **GND** - common ground for power and logic.

Signal Output

- **Sig** - this is the output signal from either the vertical or horizontal infrared receiver. When a 38KHz signal is received, the demodulated IR envelope is piped out the Signal pin into your microcontroller which will then need to decode it.

This board is specifically for receiving 38KHz IR remote control signals - it isn't going to work for proximity/distance sensing or other frequency signals.

STEMMA JST PH

- **STEMMA JST PH** (https://adafru.it/Ft4) - 2mm pitch STEMMA JST port for use with 3-pin STEMMA JST PH cables (https://adafru.it/JRA). It has connections for:
    - **GND** - common ground for power and data. It is the black wire on the JST PH cable.
    - **V+** - power input for the infrared receivers. It is the red wire on the JST PH cable.
    - **Sig** - signal to your microcontroller. It is the white wire on the JST PH cable.

Selection Switch

On the top right of the board is a slide switch labeled **V** and **H** on the board silk. This switch selects between routing the vertical IR receiver or the horizontal IR receiver to the signal (**Sig**) pin. Switch to **V** for the vertical infrared receiver and switch to **H** for the horizontal infrared receiver.

Infrared Receiver Pins

The selection switch routes either the vertical or horizontal infrared receiver to the signal pin. We can't just tie the outputs together because they'd 'fight' each other and give incoherent output. However, it's possible to read each one independently thanks to the labelled breakout pads:

- **IRH** - breakout pad for the horizontal infrared receiver
- **IRV** - breakout pad for the vertical infrared receiver

Signal LED and Jumper

- **Signal LED** - to the left of the JST PH connector is the signal LED, labeled **Sig**. It is the red LED. It will light up when an IR signal is read by the selected IR receiver.
- **LED jumper** - in the center of the back of the board is a jumper for the signal LED. It is labeled **SIG** on the board silk. If you want to disable the signal LED, cut the trace on this jumper.

Power LED and Jumper

- **Power LED** - to the right of the JST PH connector is the power LED, labeled **ON**. It is the green LED.
- **LED jumper** - in the center of the back of the board is a jumper for the power LED. It is labeled **LED** on the board silk. If you want to disable the power LED, cut the trace on this jumper.

CircuitPython and Python

It's easy to use the **Infrared IR Demodulator Breakout** with CircuitPython and the pulseio (https://adafru.it/18aP) module. This module allows you to easily write Python code for sending and receiving IR signals with IR remotes.

You can use this driver with any CircuitPython microcontroller board or with a computer that has GPIO, pulseio (https://adafru.it/19Sf) support and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library (https://adafru.it/BSN). Not all single board computers (SBCs) have pulseio support. Make sure to check if it is supported (https://adafru.it/18Ye) on your board.

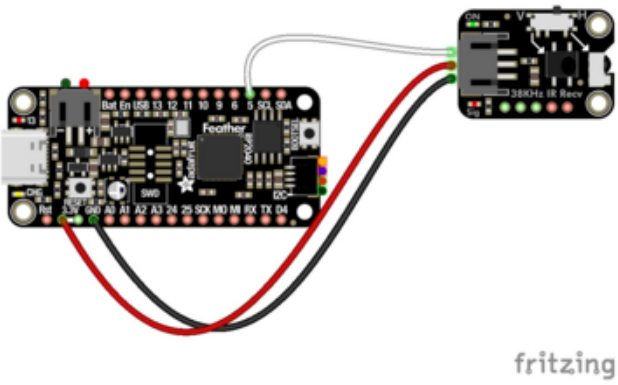
You'll need an IR remote controller to use this example with the receiver:

Mini Remote Control
This little remote control would be handy for controlling a robot or other project from across the room. It has 21 buttons and a layout we thought was handy: directional buttons and...
https://www.adafruit.com/product/389

This board is specifically for receiving 38KHz IR remote control signals - it isn't going to work for proximity/distance sensing or other frequency signals.

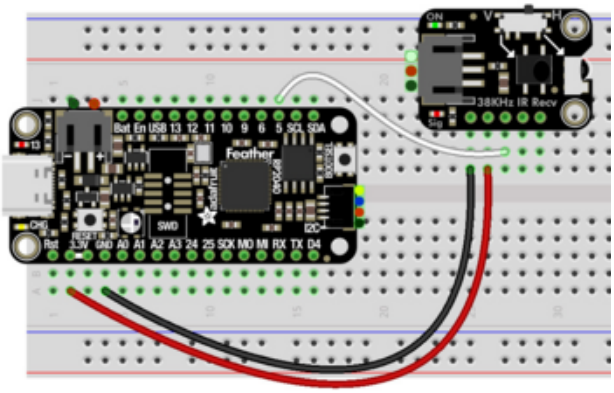
CircuitPython Microcontroller Wiring

First wire up the sensor to your board exactly as follows. The following is the receiver wired to a Feather RP2040 with a JST PH cable.



Board 3V to **receiver JST PH V+ (red wire)**
Board GND to **receiver JST PH GND (black wire)**
Board pin 5 to **receiver JST PH Sig (white wire)**

The following is the receiver wired to a Feather RP2040 using a solderless breadboard:

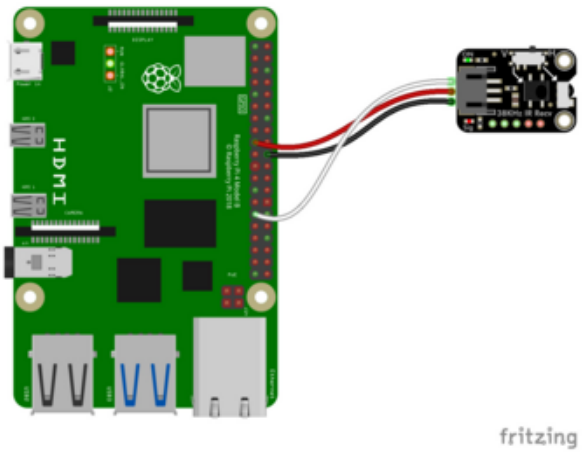


Board 3V to **receiver V+ (red wire)**
Board GND to **receiver GND (black wire)**
Board pin 5 to **receiver Sig (white wire)**

Python Computer Wiring

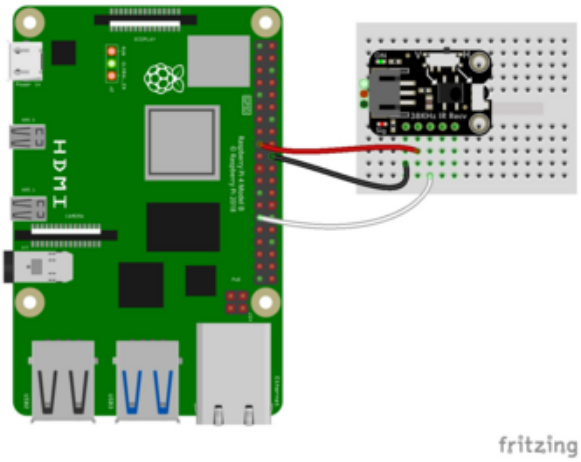
Since there are dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, [please visit the guide for CircuitPython on Linux to see whether your platform is supported](https://adafru.it/BSN) (https://adafru.it/BSN).

Here's the Raspberry Pi wired to the receiver using a JST PH cable:



Pi 3V to **receiver JST PH V+ (red wire)**
Pi GND to **receiver JST PH GND (black wire)**
Pi GPIO5 to **receiver JST PH Sig (white wire)**

Here is how you'll wire the receiver to a Raspberry Pi with a breadboard:



Pi 3V to **receiver V+ (red wire)**
Pi GND to **receiver GND (black wire)**
Pi GPIO5 to **receiver Sig (white wire)**

Python Installation of IR Remote Library

You'll need to install the **Adafruit_Blinka** library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready (https://adafru.it/BSN)!

Once that's done, from your command line run the following command:

- `pip3 install adafruit-circuitpython-irremote`

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!

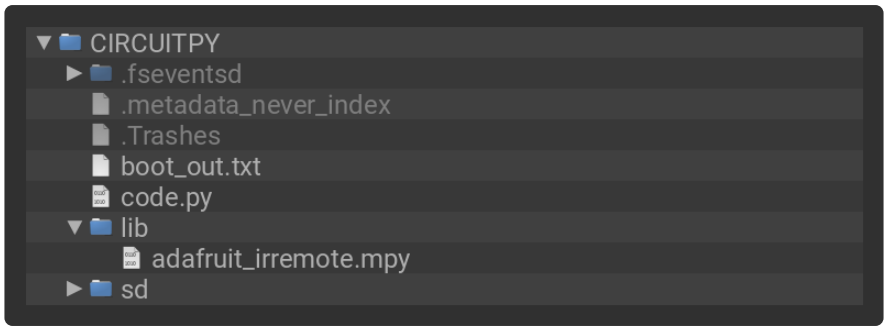
CircuitPython Usage

To use with CircuitPython, you need to first install the **Adafruit_CircuitPython_IRRemote** library into the **lib** folder on your **CIRCUITPY** drive. Then you need to update **code.py** with the example script.

Thankfully, we can do this in one go. In the example below, click the **Download Project Bundle** button below to download the necessary library and the **code.py** file in a zip file. Extract the contents of the zip file, and copy the **entire lib folder** and the **code.py** file to your **CIRCUITPY** drive.

Your **CIRCUITPY/lib** folder should contain the following file:

- **adafruit_irremote.mpy**



Python Usage

Once you have the library `pip3` installed on your computer, copy or download the following example to your computer, and run the following, replacing **code.py** with whatever you named the file:

```
python3 code.py
```

Example Code

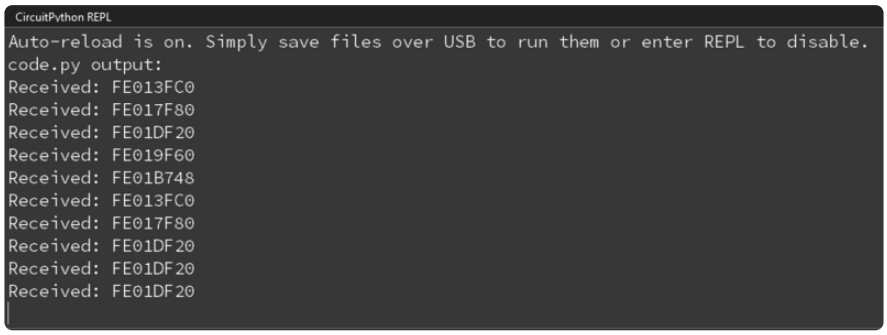
If running CircuitPython: Once everything is saved to the **CIRCUITPY** drive, connect to the serial console (https://adafru.it/Bec) to see the data printed out!

If running Python: The console output will appear wherever you are running Python.

```
# SPDX-FileCopyrightText: Copyright (c) 2024 ladyada for Adafruit Industries
#
# SPDX-License-Identifier: MIT
```

```
import board
import pulseio
import adafruit_irremote

# IR receiver setup
ir_receiver = pulseio.PulseIn(board.D5, maxlen=120, idle_state=True)
decoder = adafruit_irremote.GenericDecode()

def decode_ir_signals(p):
    codes = decoder.decode_bits(p)
    return codes

while True:
    pulses = decoder.read_pulses(ir_receiver)
    try:
        # Attempt to decode the received pulses
        received_code = decode_ir_signals(pulses)
        if received_code:
            hex_code = ''.join(["%02X" % x for x in received_code])
        print(f"Received: {hex_code}")
    except adafruit_irremote.IRNECRepeatException:  # Signal was repeated, ignore
        pass
    except adafruit_irremote.IRDecodeException:  # Failed to decode signal
        print("Error decoding")
```

In the code, if an IR remote command is received, its HEX (hexadecimal) code is printed to the serial monitor.



Python Docs

Python Docs (https://adafru.it/18aP)

Arduino

Using the Infrared IR Remote Receiver with Arduino involves wiring up the receiver to your Arduino-compatible microcontroller, installing the IRremote (https://adafru.it/18bs) library, and running the provided example code.

You'll need an IR remote controller to use this example with the receiver:

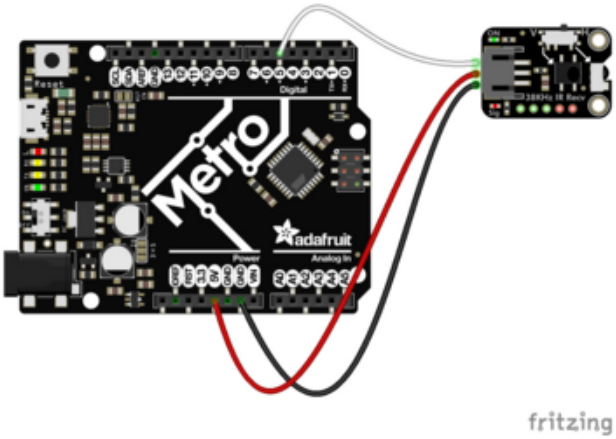
[Mini Remote Control](https://www.adafruit.com/product/389)
This little remote control would be handy for controlling a robot or other project from across the room. It has 21 buttons and a layout we thought was handy: directional buttons and...
https://www.adafruit.com/product/389

This board is specifically for receiving 38KHz IR remote control signals - it isn't going to work for proximity/distance sensing or other frequency signals.

Wiring

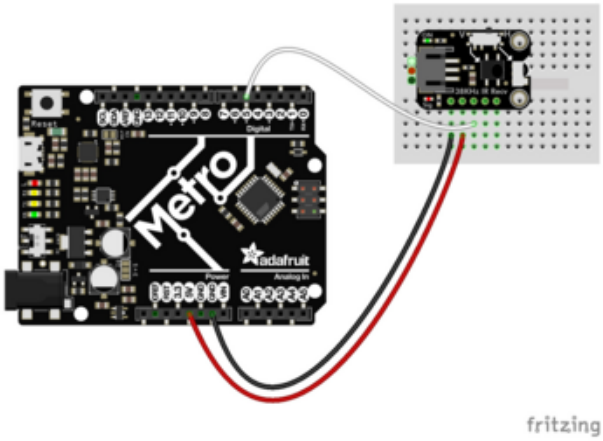
Wire as shown for a **5V** board like an Uno. If you are using a **3V** board, like an Adafruit Feather, wire the board's 3V pin to the breakout VIN.

Here is an Adafruit Metro wired up to the receiver using a JST PH cable.



Board 5V to **receiver JST PH V+ (red wire)**
Board GND to **receiver JST PH GND (black wire)**
Board pin 5 to **receiver JST PH Sig (white wire)**

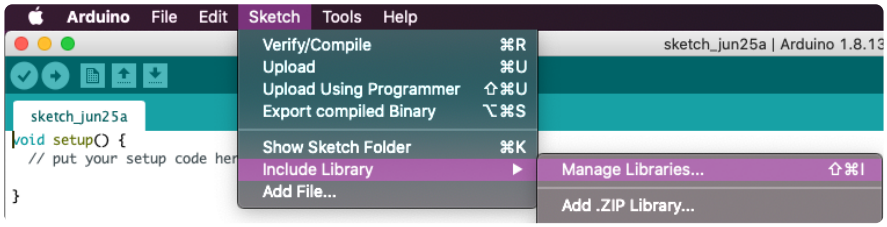
Here is an Adafruit Metro wired up using a solderless breadboard:



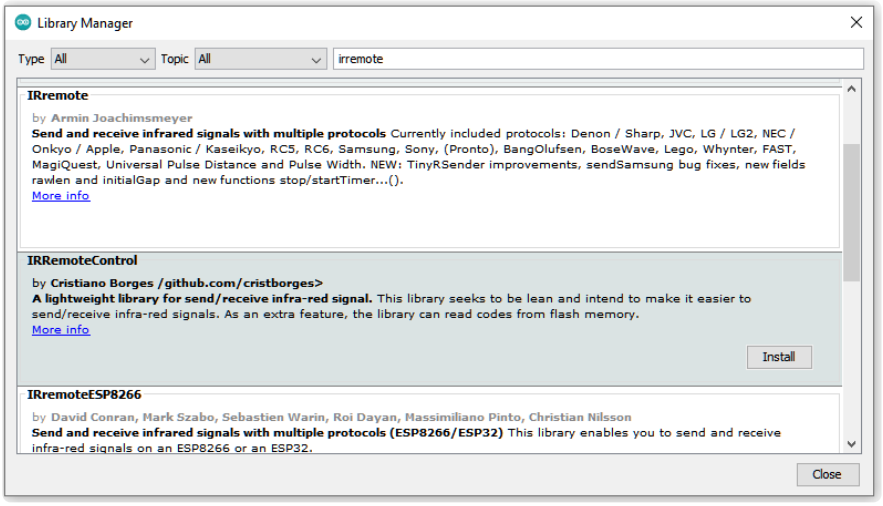
Board 5V to **receiver V+ (red wire)**
Board GND to **receiver GND (black wire)**
Board pin 5 to **receiver Sig (white wire)**

Library Installation

You can install the **IRremote** library for Arduino using the Library Manager in the Arduino IDE.



Click the **Manage Libraries ...** menu item, search for **IRremote**, and select the **IRremote** library:

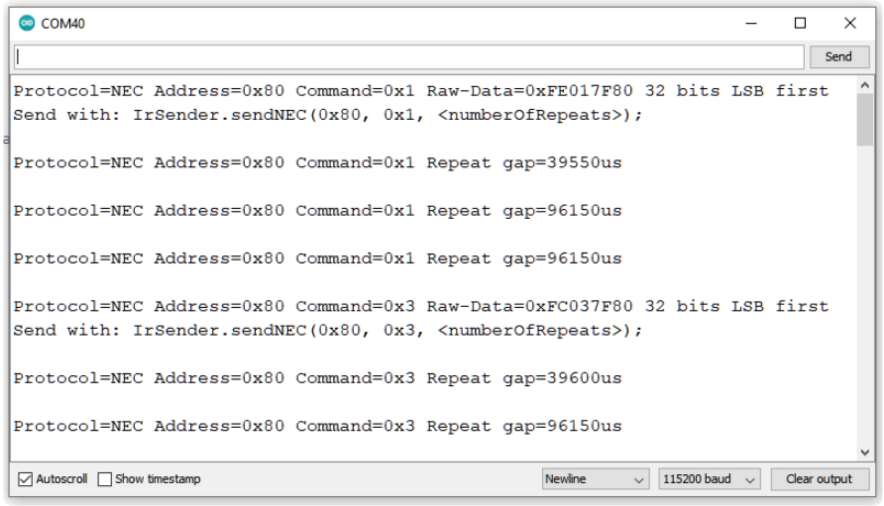


There are no additional dependencies needed for this library.

Example Code

```
// SPDX-FileCopyrightText: 2024 Liz Clark for Adafruit Industries
//
// SPDX-License-Identifier: MIT

/*
 * Based on the SimpleReceiver.cpp from the
 * Arduino-IRremote https://github.com/Arduino-IRremote/Arduino-IRremote.
 * by Armin Joachimsmeyer

************************************************************************************
 * MIT License
 *
 * Copyright (c) 2020-2023 Armin Joachimsmeyer
 *
 */

#include <Arduino.h>

#include <IRremote.hpp> // include the library
#define IR_RECEIVE_PIN   5

void setup() {
    Serial.begin(115200);
    IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK);
}

void loop() {
    /*
     * Check if received data is available and if yes, try to decode it.
     * Decoded result is in the IrReceiver.decodedIRData structure.
     *
     * E.g. command is in IrReceiver.decodedIRData.command
     * address is in command is in IrReceiver.decodedIRData.address
     * and up to 32 bit raw data in IrReceiver.decodedIRData.decodedRawData
     */
    if (IrReceiver.decode()) {
        if (IrReceiver.decodedIRData.protocol == UNKNOWN) {
            IrReceiver.printIRResultRawFormatted(&Serial, true);
            IrReceiver.resume();
        } else {
            IrReceiver.resume();
            IrReceiver.printIRResultShort(&Serial);
            IrReceiver.printIRSendUsage(&Serial);
        }
        Serial.println();
    }
}
```

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at 115200 baud. As you press buttons on your IR remote, you'll see the protocol, address, command, raw data and repeat gap print to the Serial Monitor.



Arduino Docs

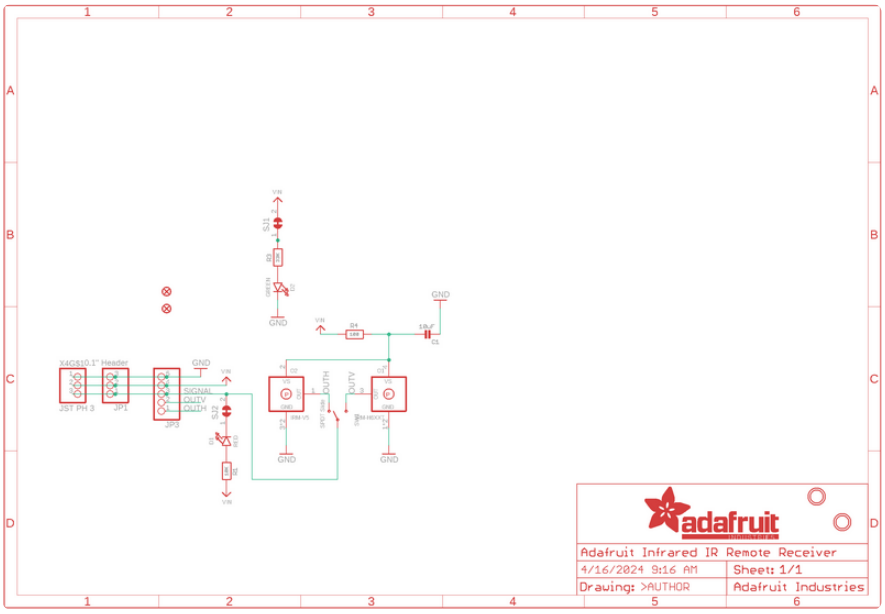
Arduino Docs (https://adafru.it/19Sa)

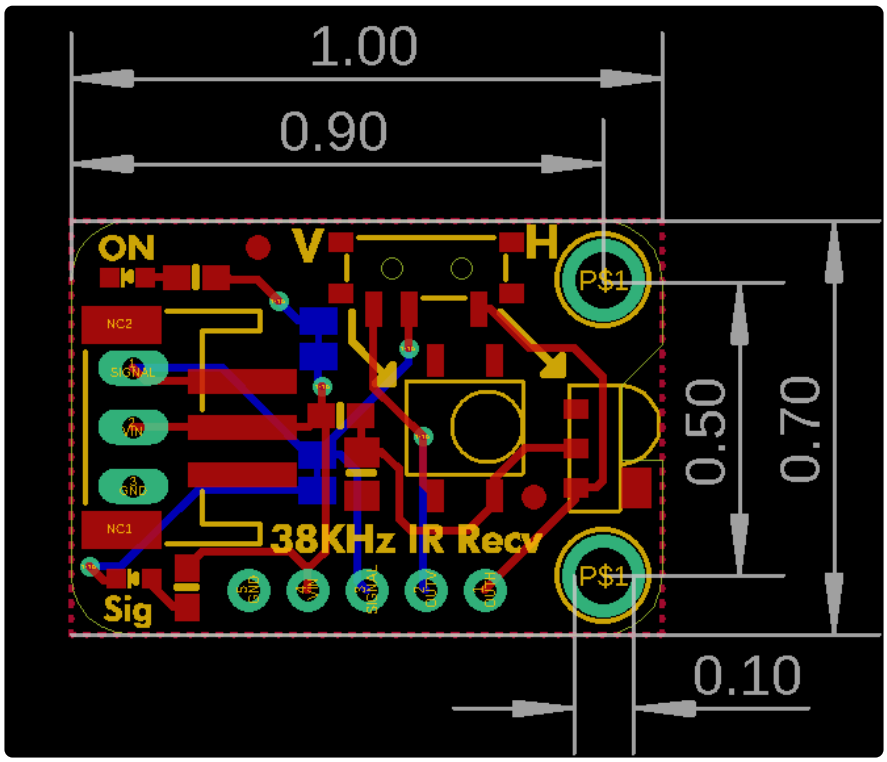
Downloads

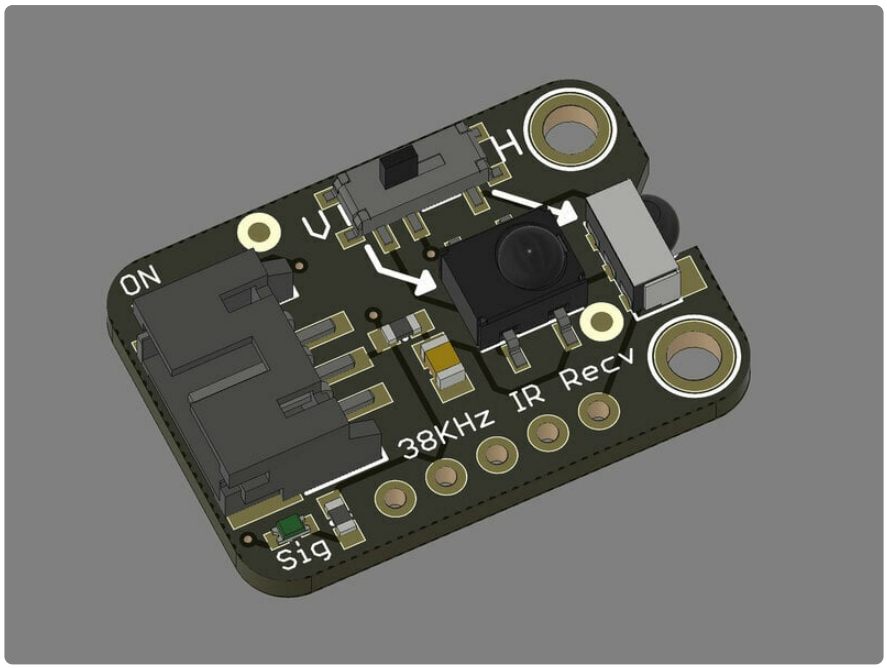
Files

- Product Demo Code (https://adafru.it/19Sb)
- EagleCAD PCB Files on GitHub (https://adafru.it/19Sc)
- 3D models on GitHub (https://adafru.it/1a15)
- Fritzing object in the Adafruit Fritzing Library (https://adafru.it/19Sd)

Schematic and Fab Print






H
V1
ON
IR Recv
38KHz
Sig