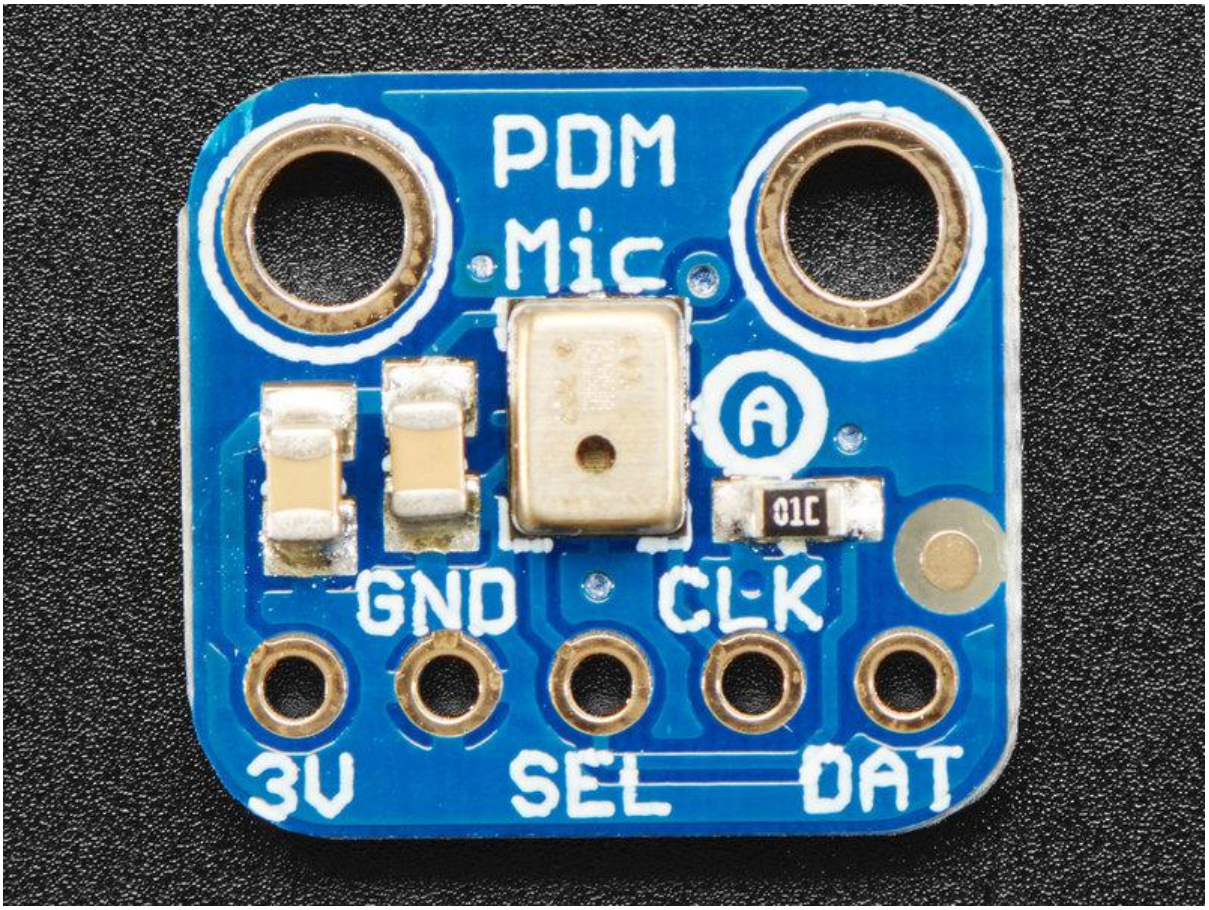
Adafruit PDM Microphone Breakout

Created by lady ada

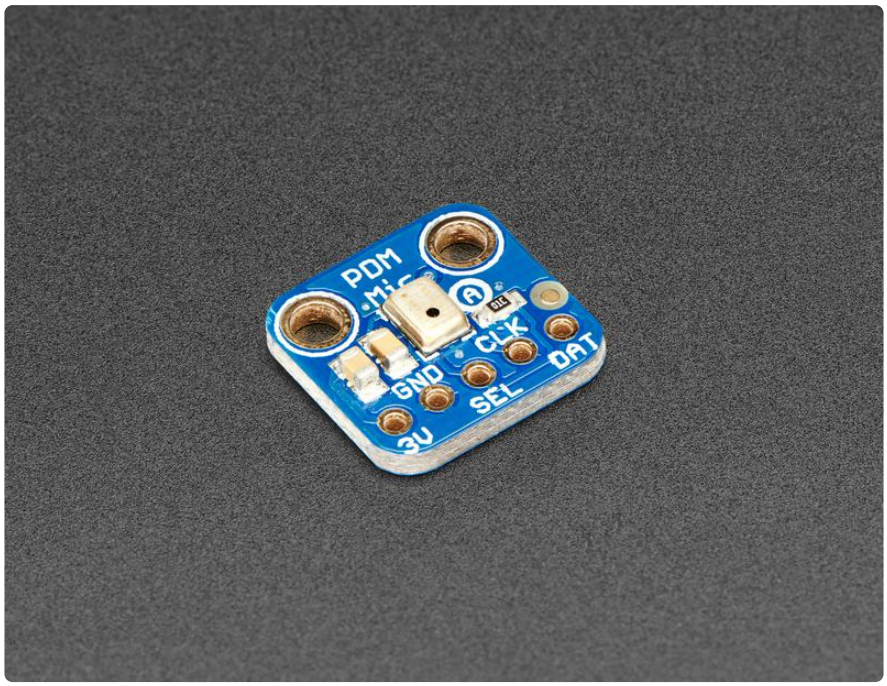


https://learn.adafruit.com/adafruit-pdm-microphone-breakout

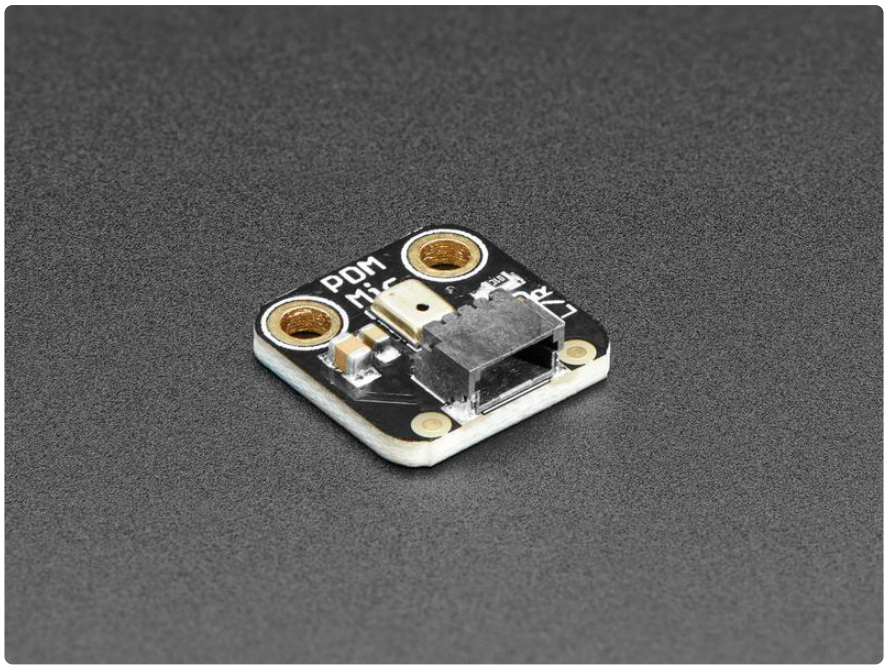
Last updated on 2022-12-01 03:09:17 PM EST

Table of Contents

Overview

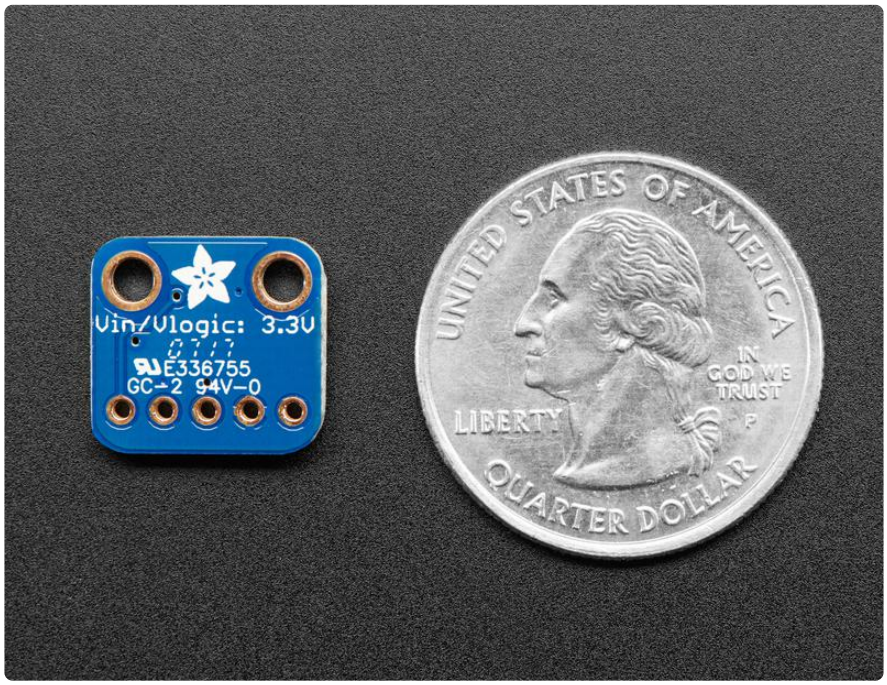


An exotic new microphone has arrived in the Adafruit shop, a PDM MEMS Microphone! PDM is the 'third' kind of microphone you can integrate with electronics, apart from analog or I2S. These microphones are very commonly used in products, but are rarely seen in maker projects. Still, they have some benefits so we thought we'd offer a breakout for the shop.

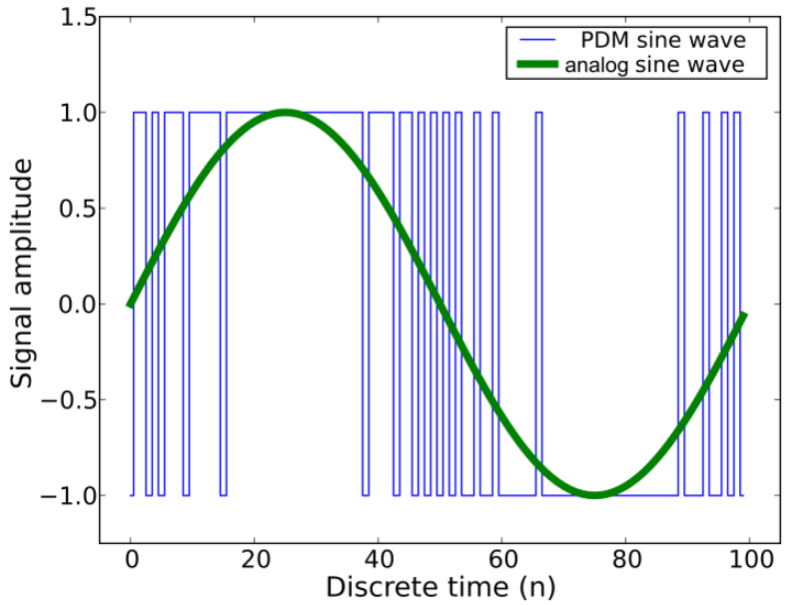


The first thing to note is that this sensor does not provide an 'analog' output like many of our electret microphone assemblies. So it's great for chips that do not have analog

inputs. Secondly, the digital interface is a very simplistic pulse density modulation output. It's digital but its not PWM and it's not I2S. You will need to make sure your chip has a PDM interface - most 32-bit processors these days do!



PDM is a little like 1-bit PWM. You clock the mic with a 1 MHz - 3 MHz clock rate, and on the data line you'll get a square wave out that syncs with the clock. The data line with be 0 or 1 logic output, with the square wave creating a density that when averaged will result in the analog value out.

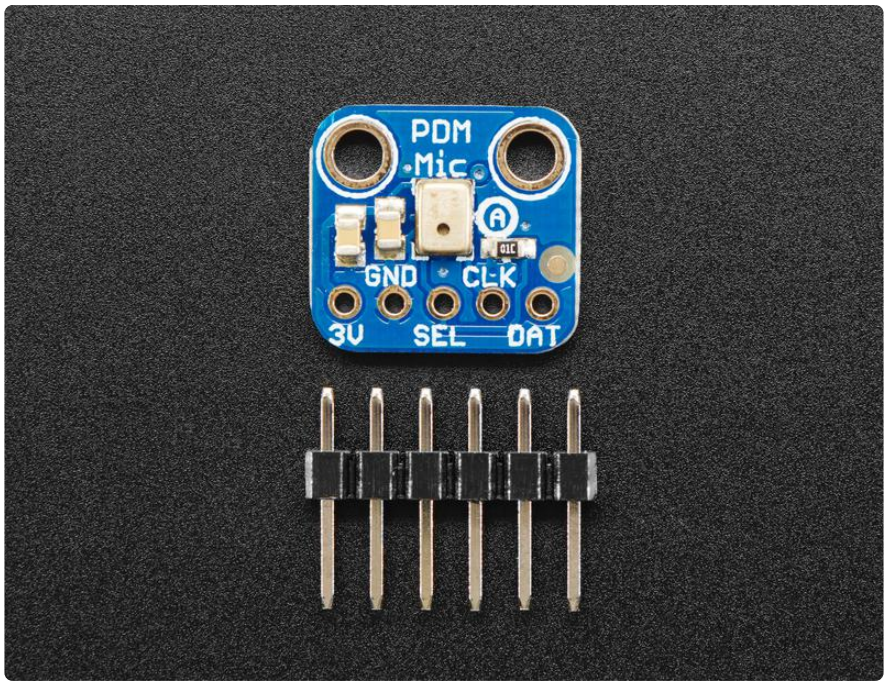


There's a few ways to manage these mics:

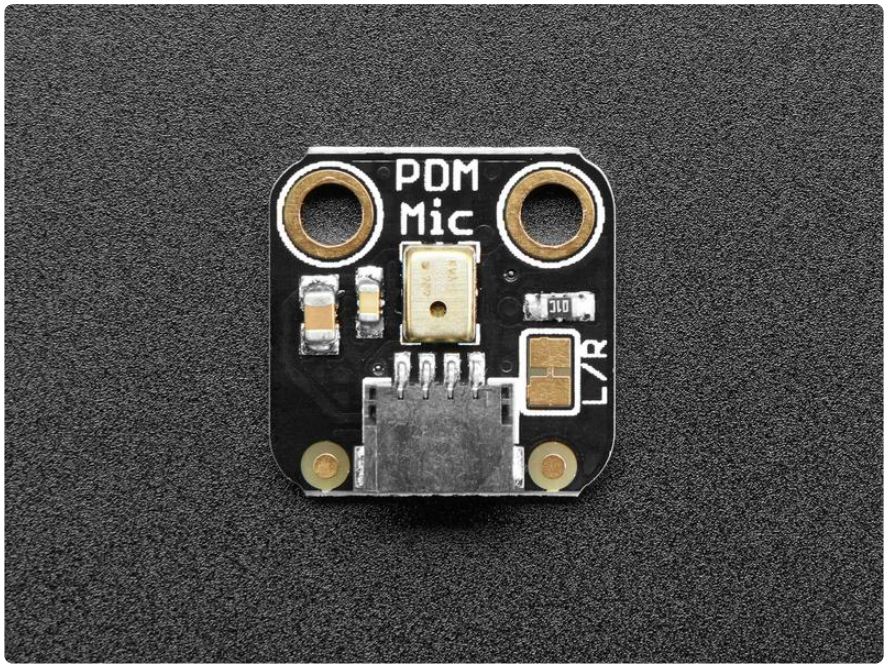
- Your chip comes with a hardware peripheral and library that does all the data managing at high speed, collects samples, applies a filter and gives you an analog value (Ideal!)
- Your chip comes with a hardware peripheral that gives you values, then it is up to you to perform the decimation/filtering. (We have some example code for this on the ATSAMD21 chipset)
- Your chip does not come with a hardware peripheral but you're pretty clever and come up with a way to make it work (See this example for the ATtiny85 ())
- You generate the high speed clock, then add an analog filter on the data line, and read the analog value (A hack, but works!)

Either way you decide to go, make sure you have a handle on what support you get with your platform, as these chips are a little tricky!

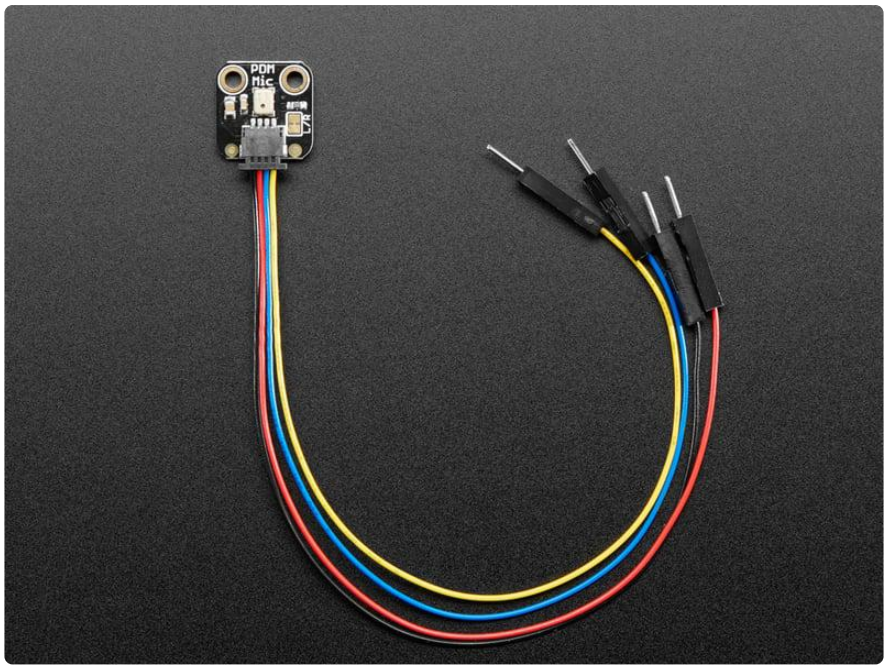
Each order of a PDM Mic comes with one fully assembled and tested microphone, and a little header to solder on for breadboard-compatibility



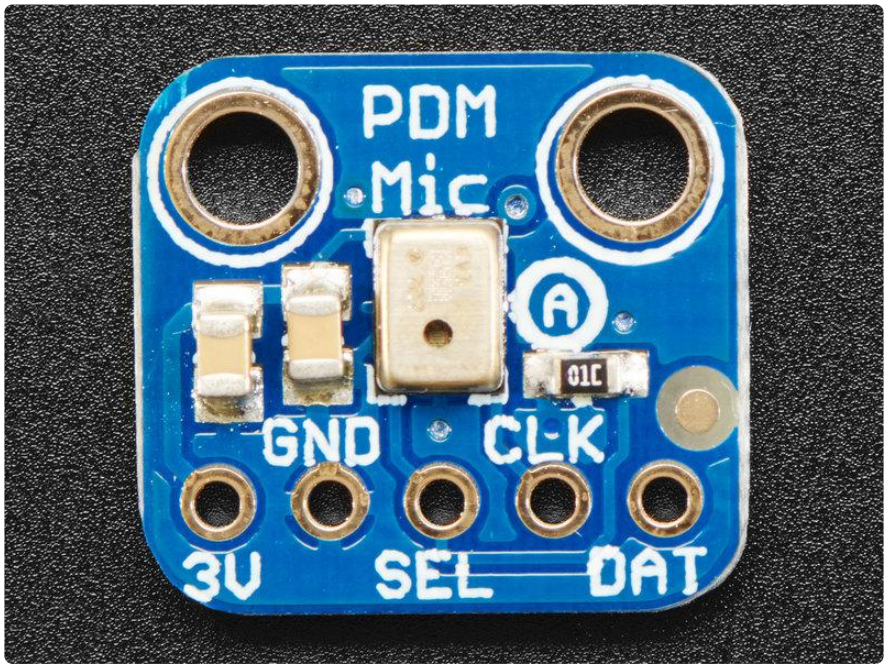
Each order of a PDM Mic with JST SH comes with one fully assembled and tested microphone. This version of the mic has a 4-JST SH connector!



The 4-pin JST connector has 3V, GND, DAT, CLK connections that can be used with one of our JST-SH cables to make a flexible mic arrangement (). If you want a version with breakout headers, we have a version here (). An on-board solder jumper lets you change the mic from Left to Right channel



Pinouts



These mics are very simple!

- 3V - This is the power input pin, this powers the chip directly. Use a quiet power supply pin if available. (The chip supports 1.8-3.3V but we have not tested it at 1.8V)
- GND - Power and data ground reference

- SEL - Left/Right select. If this pin is high, the output is on the falling edge of CLK considered the 'Right' channel. If this pin is low, the output is on the rising edge, a.k.a 'Left' channel.
- CLK - PDM clock in, 1 - 3 MHz square wave required
- DAT - PDM data out.

Arduino Wiring & Test

At this time we only have example code for the SAMD21 chipset using the I2S peripheral, you'll be limited to what pins you can use and the digital filtering must be done in software but it does work! We don't necessarily recommend this mic for SAMD21 - an analog microphone will work quite well with less hassle!

The SAMD21 can be used with PDM but it's not ideal unless you're willing to put in the work to filter data and manage the peripheral - consider this code for test & experimentation!

Available I2S Pins

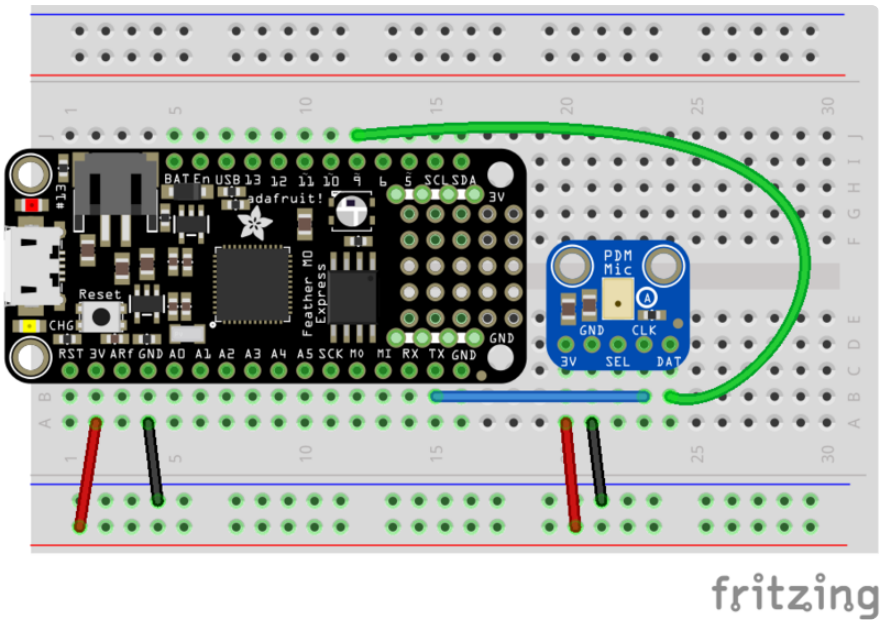
As we are using the I2S peripheral, not all pins can be used! For the Feather M0 / Metro M0 / Arduino Zero family, here's the available I2S pins:

Available Clock Pins:

- PA10 a.k.a D1 or TX
- PB11 a.k.a SCK
- PA20 a.k.a. D6

Available Data Pins:

- PA07 a.k.a D9
- PA08 a.k.a D4
- PA19 a.k.a. D12

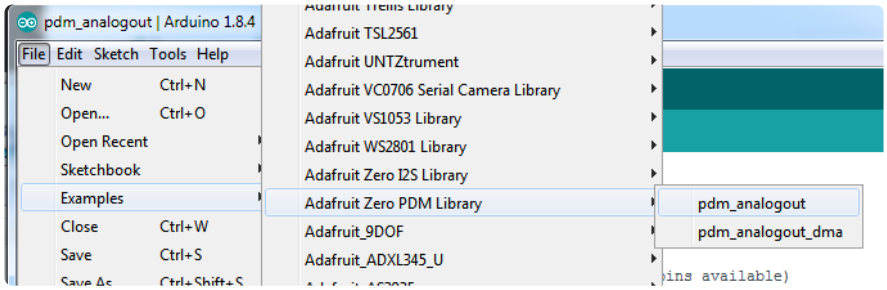


Install Library

Download the latest version of the ZeroPDM () library from github. Install as usual!

We have a two examples, one uses the DMA capability to grab data, which means we don't have to do as much work, but there's more setup involved and requires the ZeroDMA library ().

We recommend starting with the basic demo, which will echo audio data to A0 (the the analog output). Connect up headphones or an oscilloscope to A0 to hear/see the audio!



Before uploading, be sure to change the instantiator to match your pinouts:

```
// Create PDM receiver object, with Clock and Data pins used (not all pins
available)
Adafruit_ZeroPDM pdm = Adafruit_ZeroPDM(1, 4);  // Metro M0 or Arduino zero
//Adafruit_ZeroPDM pdm = Adafruit_ZeroPDM(34, 35);  // CPlay express
```

CircuitPython

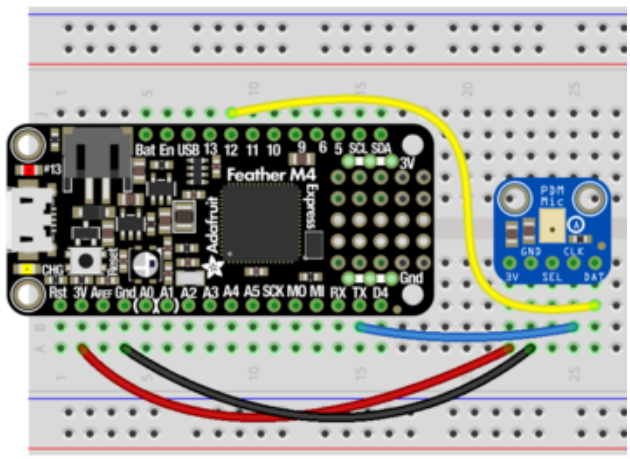
It's easy to use the Adafruit PDM microphone breakout with CircuitPython, using the built-in `audiobusio` module () and `PDMIn` class (). It allows you to record an input audio signal from the microphone using PDM.

This page will walk you through wiring up your PDM mic, and using it to print out sound levels to the serial console and show the values on the plotter in Mu.

CircuitPython Microcontroller Wiring

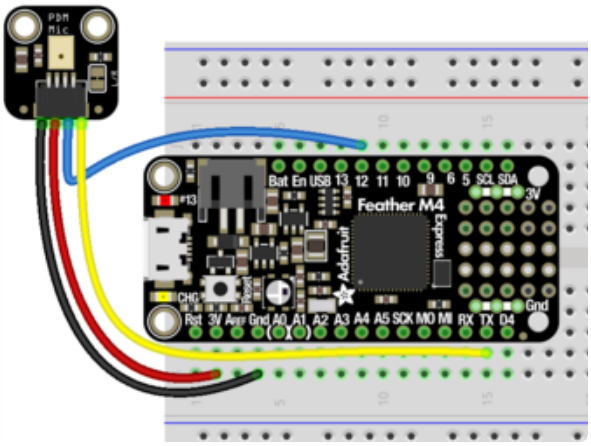
The following wiring diagrams show how to connect the PDM mic to a Feather M4 Express. If you're using another board, check out the Where's my PDMIn? section () at the end for valid pin combinations for your board. Some boards, like the SAMD21 and SAMD51 have fixed pins that support PDM. Others like the nRF52840 can use any two pins.

The following is the header version of the PDM microphone breakout wired to a Feather M4 Express:



Mic 3V to Feather 3V
Mic GND to Feather Gnd
Mic CLK to Feather TX
Mic DAT to Feather D12

The following is the JST version of the PDM microphone breakout wired to a Feather M4 Express:



Mic 3V (red wire) to Feather 3V
Mic GND (black wire) to Feather Gnd
Mic DAT (blue wire) to Feather D12
Mic CLK (yellow wire) to Feather TX

If you're using Circuit Playground Express, there is a built in PDM microphone. There is a guide page dedicated to using Circuit Playground Express and the built-in microphone (). If you're using a CPX, check that out instead!

CircuitPython Usage

As `PDMIn` is built into CircuitPython, no separate libraries are necessary for this example!

Save the following as code.py on your microcontroller board:

```
# SPDX-FileCopyrightText: 2018 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import time
import array
import math
import board
import audiobusio


# Remove DC bias before computing RMS.
def mean(values):
    return sum(values) / len(values)


def normalized_rms(values):
    minbuf = int(mean(values))
    samples_sum = sum(
        float(sample - minbuf) * (sample - minbuf)
        for sample in values
    )

    return math.sqrt(samples_sum / len(values))


# Main program
mic = audiobusio.PDMIn(board.TX, board.D12, sample_rate=16000, bit_depth=16)
samples = array.array('H', [0] * 160)
```

```
while True:
    mic.record(samples, len(samples))
    magnitude = normalized_rms(samples)
    print((magnitude,))
    time.sleep(0.1)
```

First you import `time`, `array`, `math`, `board` and `audiobusio`.

Then you have two helper functions. The first one uses math to return a mean, or average. It is used in the second helper. The second one uses math to return a normalised RMS average (). You use these functions to take multiple sound samples really quickly and average them to get a more accurate reading.

Next you set up the microphone object and your samples variable.

Then you use the mic object to start taking sound samples. You use the normalised RMS to find the average of a given set of samples, and you call that the `magnitude`. Last, you `print` the `magnitude` to the serial console.

Note that the Mu plotter looks for tuple values to print. Tuples in Python come in parentheses `()` with comma separators. If you have two values, a tuple would look like `(1.0, 3.14)` Since you have only one value, you need to have it print out like `(1.0,)` note the parentheses around the number, and the comma after the number. Thus the extra parentheses and comma in `print((magnitude,))`.

Once you have everything setup and running, try speaking towards the microphone, and watch the plotter immediately react! Move further away from the microphone to cause smaller changes in the plotter line. Move closer to the board to see bigger spikes!

Note that the way that the code works with averaging a given number of readings over time means that short sounds like claps can sometimes get missed. If you feel like your mic is not responding, try a longer sound like a hum or speaking words.

It's a really easy way to test your microphone and see how it reads sound changes!

Where's my `PDMIn`?

Save the following as code.py on your board, and connect to the serial console to see a list of all the valid `PDMIn` pin combinations. Note: the code will run immediately and only runs once - if you connect to the serial console and don't see anything, press ctrl+D to reload and run the code again.

```
# SPDX-FileCopyrightText: 2018 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

import board
import audiobusio
from microcontroller import Pin


def is_hardware_PDM(clock, data):
    try:
        p = audiobusio.PDMIn(clock, data)
        p.deinit()
        return True
    except ValueError:
        return False
    except RuntimeError:
        return True


def get_unique_pins():
    exclude = ['NEOPIXEL', 'APA102_MOSI', 'APA102_SCK']
    pins = [pin for pin in [
        getattr(board, p) for p in dir(board) if p not in exclude]
            if isinstance(pin, Pin)]
    unique = []
    for p in pins:
        if p not in unique:
            unique.append(p)
    return unique


for clock_pin in get_unique_pins():
    for data_pin in get_unique_pins():
```

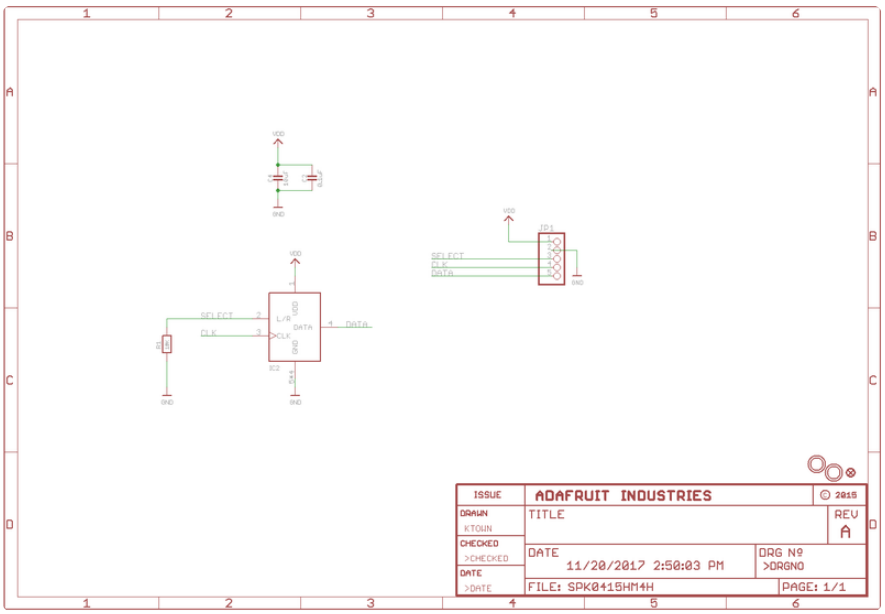
```
        if clock_pin is data_pin:
            continue
        if is_hardware_PDM(clock_pin, data_pin):
            print("Clock pin:", clock_pin, "\t Data pin:", data_pin)
        else:
            pass
```

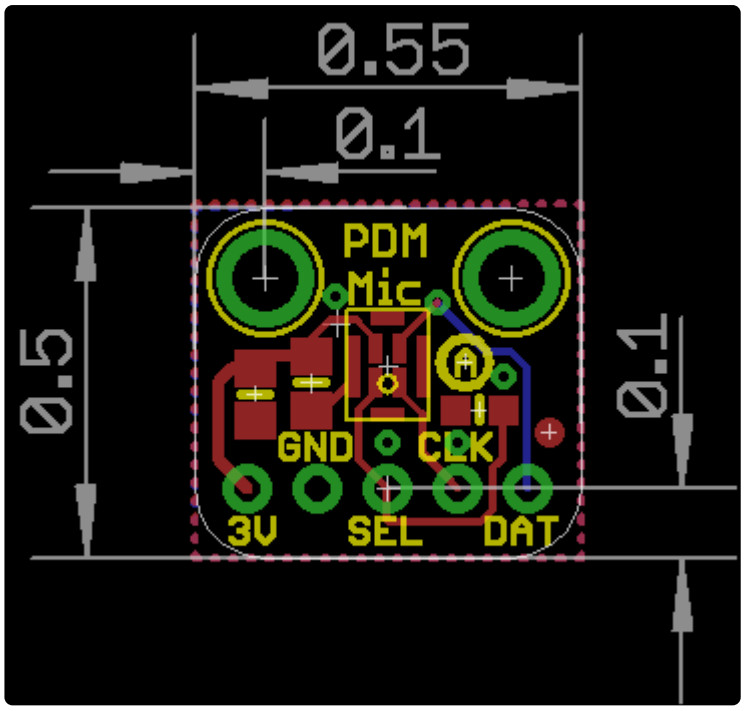
Downloads

Files

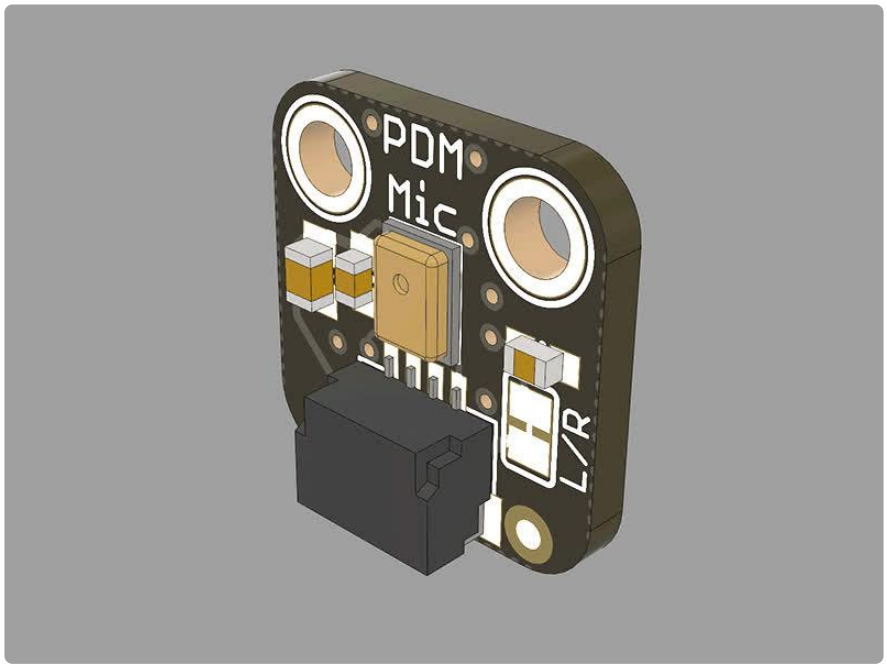
- MP34DT01-M datasheet ()
- Fritzing file on GitHub ()
- 3D Models on GitHub ()

Schematic & Fabrication Print





3D Model



PDM Mic with JST connector