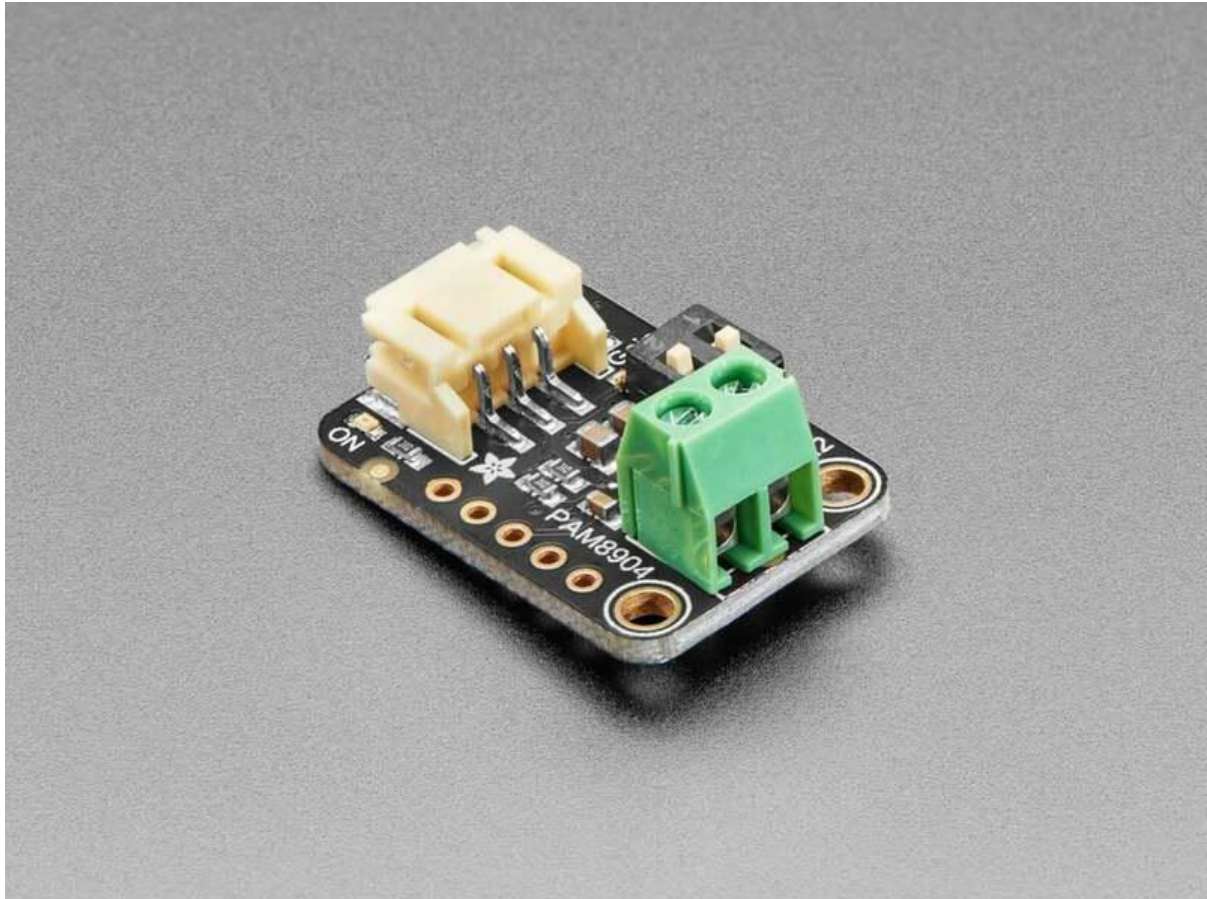




Adafruit STEMMA Piezo Driver Amp

Created by Liz Clark



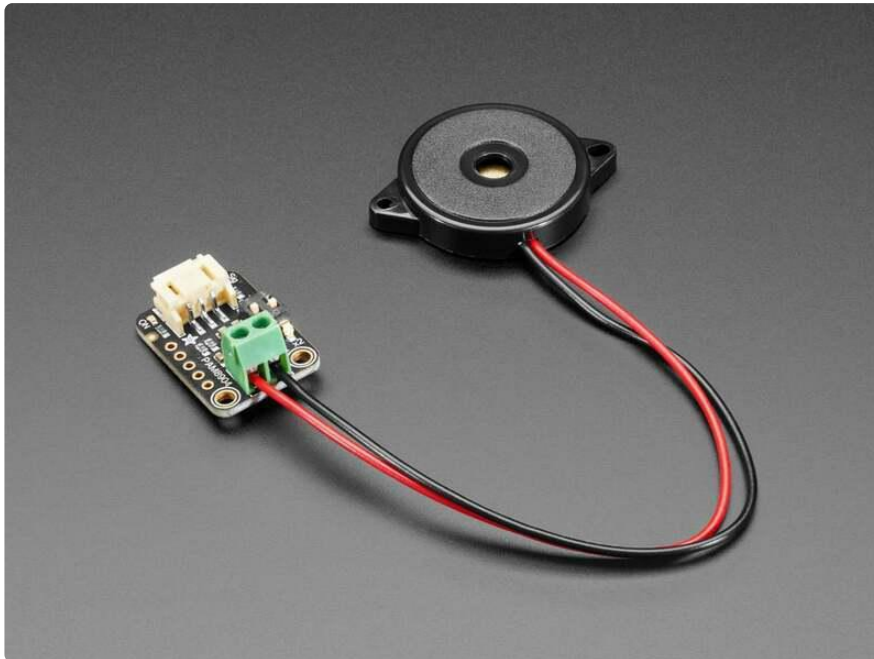
<https://learn.adafruit.com/adafruit-stemma-piezo-driver-amp>

Last updated on 2023-09-06 12:01:39 PM EDT

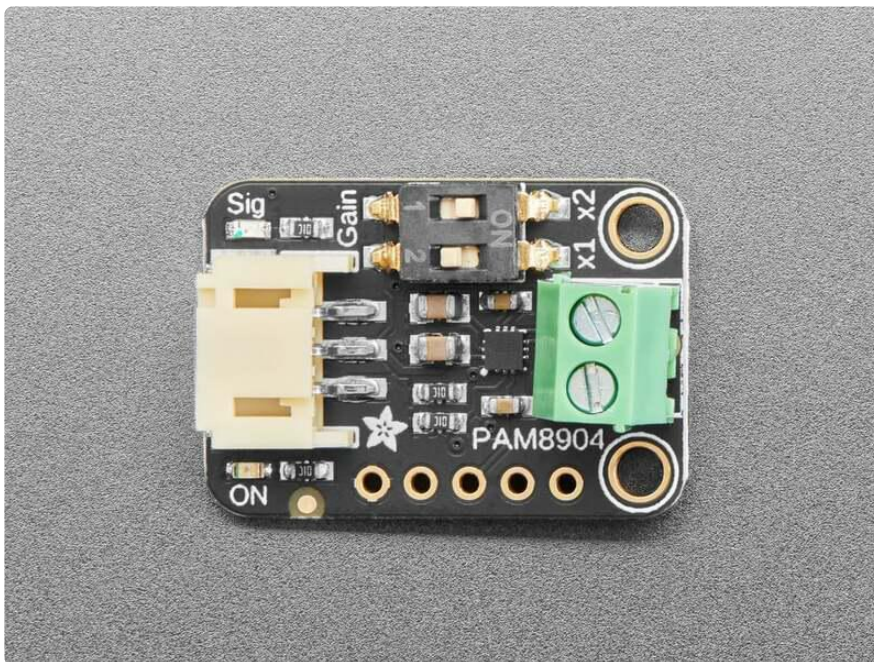
Table of Contents

Overview	3
Pinouts	5
<ul style="list-style-type: none">• Power Pins• I/O Pins• STEMMA Connector• Terminal Block• Gain Selector Switch• Power LED and Jumper• Signal LED and Jumper	
CircuitPython and Python	7
<ul style="list-style-type: none">• CircuitPython Microcontroller Wiring• Python Computer Wiring• Python Setup• CircuitPython Usage• Python Usage• Example Code	
Python Docs	11
Arduino	11
<ul style="list-style-type: none">• Wiring• Example Code	
Arduino Docs	13
Downloads	13
<ul style="list-style-type: none">• Files• Schematic and Fab Print	

Overview



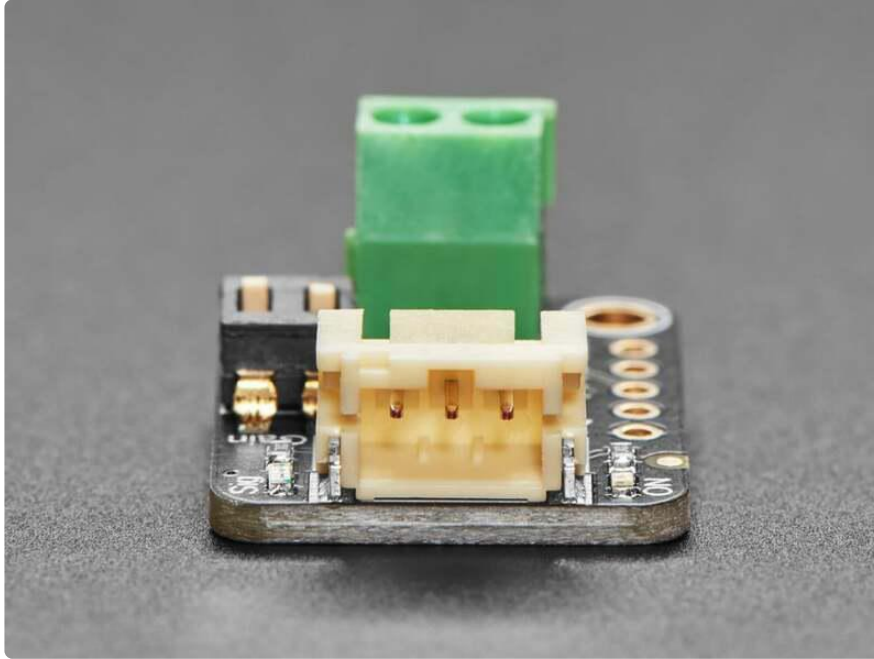
Piezos make noise when you put an AC voltage across them - and the bigger the voltage, the louder they are. With your standard 3V logic microcontroller you can make 3 volts peak to peak (Vpp) with a PWM out, or 6Vpp differential with two complementary outputs. But what if you want even louder? Or if you're using a piezo to sense distance using ultrasonic bounces?



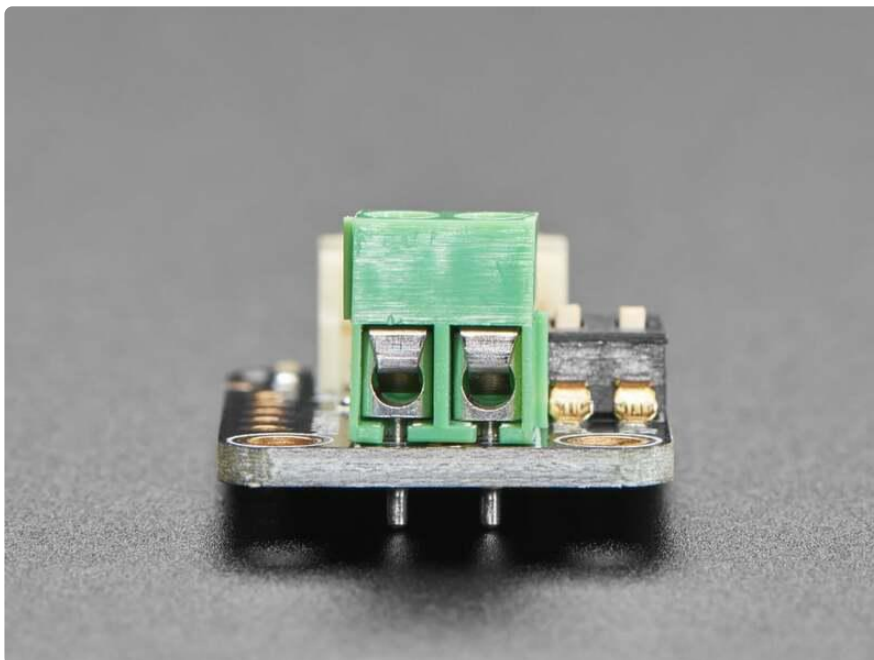
We found the nifty [PAM8904 \(\)](#), which is an amplifier specifically designed for driving piezo elements - and unlike audio amplifiers, it's good for up to 300 KHz! It's a

switched-cap piezo driver that has bridge-tied load (BTL) output and up to 3x voltage multiplication thanks to a built-in boosting circuit for up to ~13Vpp.

We whipped up a quick breakout in our 2mm JST-PH STEMMA form-factor to make it easy for anyone who wants to beep their boops very loudly.



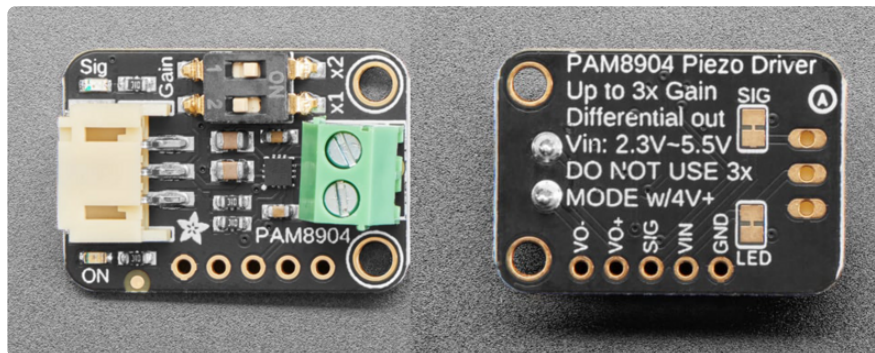
Usage is easy: power with 3 to 5VDC on the Vin and GND pins. Then provide a square wave on the signal pin, from 20Hz to 300KHz - any duty cycle is ok but 50% will probably work best. There's a dual DIP switch to set the gain: you can set it to off (zero gain), x1 gain, x2 gain and x3 gain. The output is differential, so 2x gain will give you 4xVin peak-to-peak across the piezo lines. Connect your piezo to the terminal block and you're ready to rock.



Please note: If you are powering the driver from 5VDC, don't set the gain to x3 because the 15V output is higher than the driver is specified for. (Yeah we also sorta wondered why the chip manufacturer allows it). So if you are using 3.3V power, x3 gain is OK and will give you about 10V out but if you're powering with 5V use 2x gain max to keep the max voltage at 10V.

[Piezo element \(\)](#) and [3-pin JST PH cable \(\)](#) not included! But we stock 'em in the shop if you want to pick up separately.

Pinouts



Power Pins

- VIN - This is the power pin - provide between 2.3 to 5V DC
- GND - This is common ground for power and logic.

I/O Pins

- SIG - The signal input pin. Provide a square wave from 20Hz to 300KHz to output to the attached piezo. The signal level does not have to match the power voltage level
- VO+ - The positive output for a piezo.
- VO- - The negative output for a piezo.

STEMMA Connector

The STEMMA connector is located on the left side of the front of the board, to the left of the Gain DIP switch. You can use a STEMMA JST PH 2mm 3-pin cable to connect the driver to your microcontroller board, for example [this cable \(\)](#) to connect using a breadboard.

The STEMMA connector has the following pins:

- SIGNAL (white wire) - Signal input pin
- VIN (red wire) - Power pin, 3-5VDC
- GND (black wire) - Ground pin.

Terminal Block

The terminal block is located on the right side of the front of the board, above the PA M8904 label on the board silk.

The block has the following terminals:

- VO+ - This terminal is towards the top of the board, and is located next to the x1 label on the board silk. It is the same as the VO+ pin. Connect the positive wire on your piezo to this terminal.
- VO- - This terminal is towards the bottom of the board and is located next to the PAM8904 label on the board silk. It is the same as the VO- pin. Connect the negative wire on your piezo to this terminal.

Gain Selector Switch

- Gain - The dual DIP switch at the top of the board lets you adjust the gain output from the PAM8904. You can set it to off (zero gain), x1 gain by flipping the x1 switch to ON, x2 gain by flipping the x2 switch to ON and x3 gain by flipping both the x1 and x2 switches to ON. The output is differential so 2x gain will give you 4xVin peak-to-peak across the piezo lines.

Please note: If you are powering the driver from 5VDC, don't set the gain to x3 because the 15V output is higher than the driver is specified for. So if you are using 3.3V power, x3 gain is OK and will give you about 10V out but if you're powering with 5V use 2x gain max to keep the max voltage at 10V.

If you are powering the driver from 5VDC, don't set the gain to x3 because the 15V output is higher than the driver is specified for.

Power LED and Jumper

- Power LED - On the front of the board, below the JST-PH connector, is the power LED, labeled ON. It is a green LED.
- Power LED jumper - In the lower right corner on the back of the board is a jumper for the power LED. It is labeled LED on the board silk. To disable the power LED, cut the trace on this jumper. To enable it again, solder the pads back together.

Signal LED and Jumper

- Signal LED - On the front of the board, above the JST-PH connector, is the signal LED, labeled Sig. It is a red LED. It will light-up every time a signal is received on the signal pin.
- Signal LED jumper - In the upper right corner on the back of the board is a jumper for the signal LED. It is labeled SIG on the board silk. To disable the signal LED, cut the trace on this jumper. To enable it again, solder the pads back together.

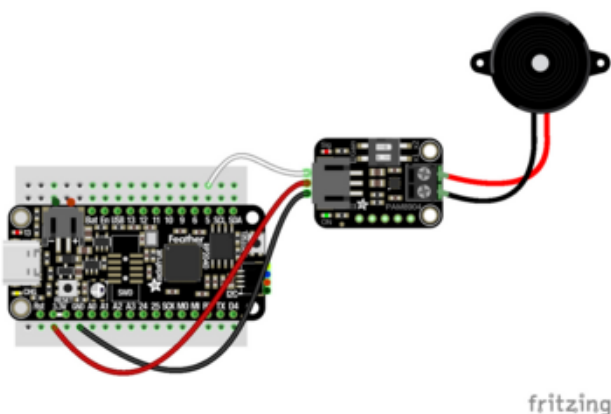
CircuitPython and Python

It's easy to use the STEMMA Piezo Driver Amp with Python or CircuitPython, and the [pwmiio \(\)](#) module. This module allows you to easily write Python code to control pulse width modulation (PWM).

You can use this driver with any CircuitPython microcontroller board or with a computer that has GPIO and Python [thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library \(\)](#).

CircuitPython Microcontroller Wiring

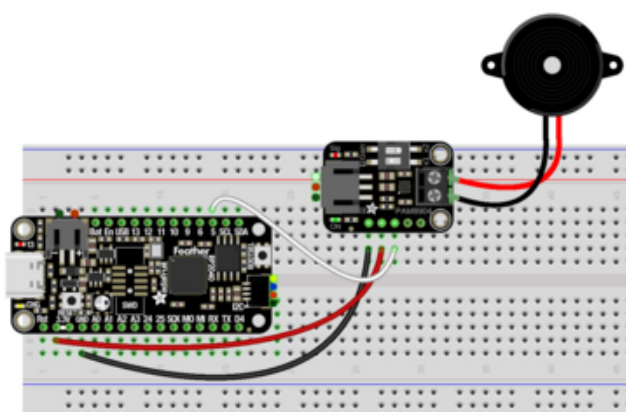
First wire up a driver to your board exactly as follows along with a piezo element. The following is the driver wired to a Feather RP2040 using the STEMMA JST-PH connector:



- Board 3.3V to driver VIN (red wire)
- Board GND to driver GND (black wire)
- Board pin 5 to driver SIG (white wire)
- Driver VO+ to piezo positive (red wire)
- Driver VO- to piezo negative (black wire)

fritzing

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

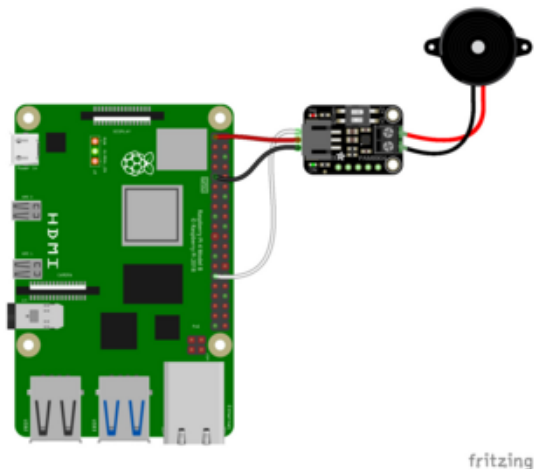


- Board 3.3V to driver VIN (red wire)
- Board GND to driver GND (black wire)
- Board pin 5 to driver SIG (white wire)
- Driver VO+ to piezo positive (red wire)
- Driver VO- to piezo negative (black wire)

Python Computer Wiring

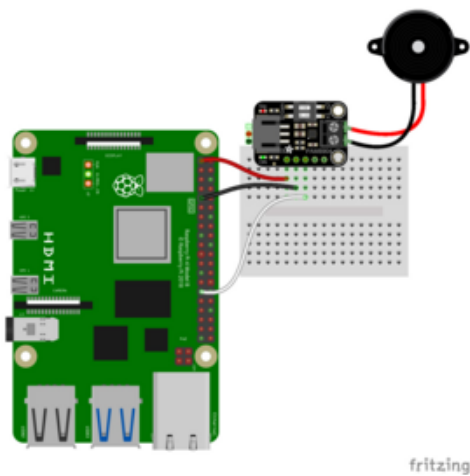
Since there are dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, [please visit the guide for CircuitPython on Linux to see whether your platform is supported \(\)](#). [pwmio \(\)](#) will also need to be implemented for your board.

Here's the Raspberry Pi wired using the STEMMA JST-PH connector:



Pi 3.3V to driver VIN (red wire)
Pi GND to driver GND (black wire)
Pi GPIO 5 to driver SIG (white wire)
Driver VO+ to piezo positive (red wire)
Driver VO- to piezo negative (black wire)

Here's the Raspberry Pi using a solderless breadboard:



Pi 3.3V to driver VIN (red wire)
Pi GND to driver GND (black wire)
Pi GPIO 5 to driver SIG (white wire)
Driver VO+ to piezo positive (red wire)
Driver VO- to piezo negative (black wire)

Python Setup

You'll need to install the Adafruit_Blinka library that provides the CircuitPython support in Python. This may also require 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 \(\)!](#)

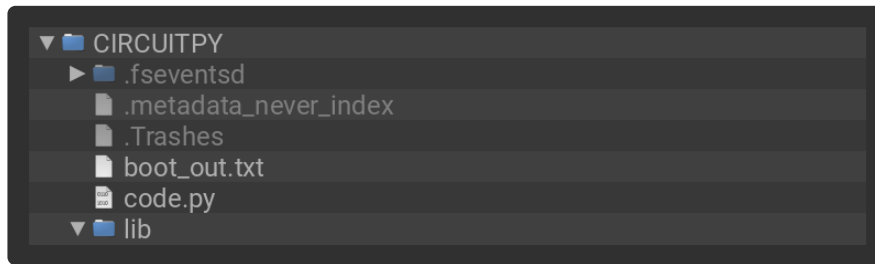
`pwmio` is a built-in module and does not need to be installed separately using `pip`. Make sure to check that `pwmio` is supported on your platform though if you aren't using something more common like a Raspberry Pi.

CircuitPython Usage

To use with CircuitPython, 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 `code.py` file in a zip file. Extract the

contents of the zip file, and copy the code.py file to your CIRCUITPY drive. The example uses built-in modules, so no additional libraries need to be copied to the CIRCUITPY/lib folder.



Python Usage

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

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

import time
import board
import pwmio

piezo = pwmio.PWMOut(board.D5, duty_cycle=0, frequency=440, variable_frequency=True)

while True:
    for f in (262, 294, 330, 349, 392, 440, 494, 523):
        piezo.frequency = f
        piezo.duty_cycle = 65535 // 2 # On 50%
        time.sleep(0.25) # On for 1/4 second
        piezo.duty_cycle = 0 # Off
        time.sleep(0.05) # Pause between notes
    time.sleep(0.5)
```

When you run the example code, you'll hear an ascending C major scale through the piezo element on a loop. Use the gain DIP switch to adjust the gain for the piezo element.

Python Note

If you run this example on a Raspberry Pi, you'll see this printed to the serial console when the code starts running:

Variable Frequency is not supported, continuing without it...

After that you'll hear the tones being played through the piezo element.

Python Docs

[Python Docs \(\)](#)

Arduino

Using the STEMMA Piezo Driver Amp with Arduino involves wiring up the adapter to your Arduino-compatible microcontroller, attaching a piezo element to the driver and running the provided example code.

Wiring

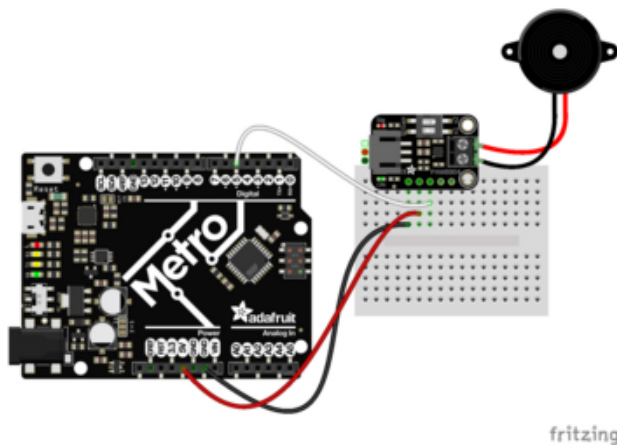
Wire as shown for a 5V board like an Uno. If you are using a 3V board, like an Adafruit Feather, wire the board's 3V pin to the driver VIN.

Here is an Adafruit Metro wired up to the driver using the STEMMA JST-PH connector:



Board 5V to driver VIN (red wire)
Board GND to driver GND (black wire)
Board pin 5 to driver SIG (white wire)
Driver VO+ to piezo positive (red wire)
Driver VO- to piezo negative (black wire)

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



- Board 5V to driver VIN (red wire)
- Board GND to driver GND (black wire)
- Board pin 5 to driver SIG (white wire)
- Driver VO+ to piezo positive (red wire)
- Driver VO- to piezo negative (black wire)

Please note: If you are powering the driver from 5VDC, don't set the gain to x3 because the 15V output is higher than the driver is specified for. If you're powering with 5V use 2x gain max to keep the max voltage at 10V.

If you are powering the driver from 5VDC, don't set the gain to x3 because the 15V output is higher than the driver is specified for.

Example Code

```
// SPDX-FileCopyrightText: 2023 Liz Clark for Adafruit Industries
//
// SPDX-License-Identifier: MIT

#define PIEZO_PIN 5 // Pin connected to the piezo buzzer.

int toneFreq[] = {262, 294, 330, 349, 392, 440, 494, 523};
int toneCount = 8;

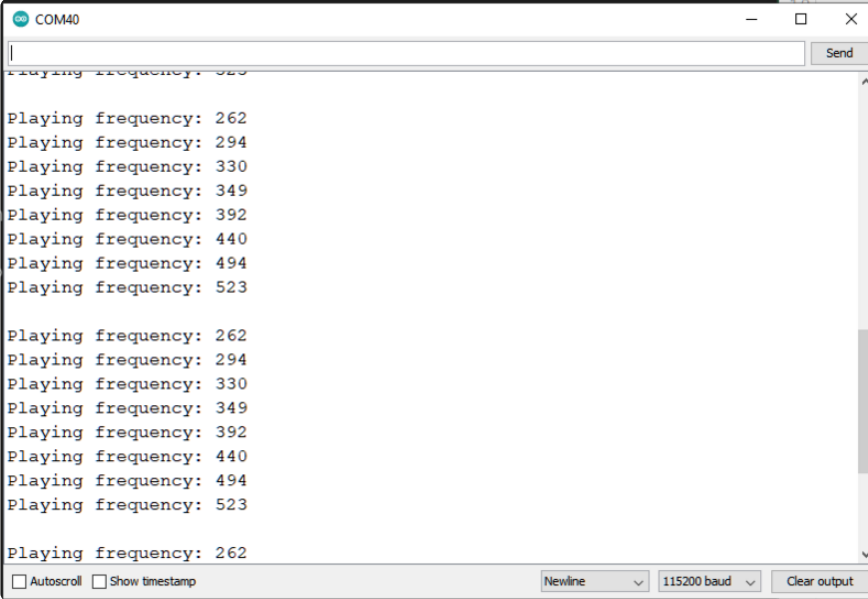
void setup() {
  Serial.begin(115200);
  Serial.println("Piezo Tone Example");
}

void loop() {

  for (int i=0; i < toneCount; i++) {
    Serial.print("Playing frequency: ");
    Serial.println(toneFreq[i]);
    tone(PIEZO_PIN, toneFreq[i]);
    delay(250); // Pause for half a second.
    noTone(PIEZO_PIN);
    delay(50);
  }
  Serial.println();
  delay(1000);
}
```

Upload the sketch to your board and open up the Serial Monitor (Tools -> Serial Monitor) at 115200 baud. You'll see the tone frequencies printed to the Serial Monitor

as they are played through the piezo element. Use the gain DIP switch to adjust the gain for the piezo element.



```
COM40
Playing frequency: 262
Playing frequency: 294
Playing frequency: 330
Playing frequency: 349
Playing frequency: 392
Playing frequency: 440
Playing frequency: 494
Playing frequency: 523

Playing frequency: 262
Playing frequency: 294
Playing frequency: 330
Playing frequency: 349
Playing frequency: 392
Playing frequency: 440
Playing frequency: 494
Playing frequency: 523

Playing frequency: 262
```

Arduino Docs

[Arduino Docs \(\)](#)

Downloads

Files

- [PAM8904 Datasheet \(\)](#)
- [EagleCAD PCB Files on GitHub \(\)](#)
- [Fritzing object in the Adafruit Fritzing Library \(\)](#)

Schematic and Fab Print

