

# Instructions:



## AMPLIFIED SPEAKER 2



Instructions version 1b.

# TABLE OF CONTENTS

Introduction .....	3
Downloading the Examples.....	4
Raspberry Pi 1-4.....	5
Raspberry Pi Pico and Pico W.....	6
ESP32.....	9
Arduino.....	11
Troubleshooting.....	13
Support.....	14
Books.....	15
MonkMakes.....	16

# INTRODUCTION

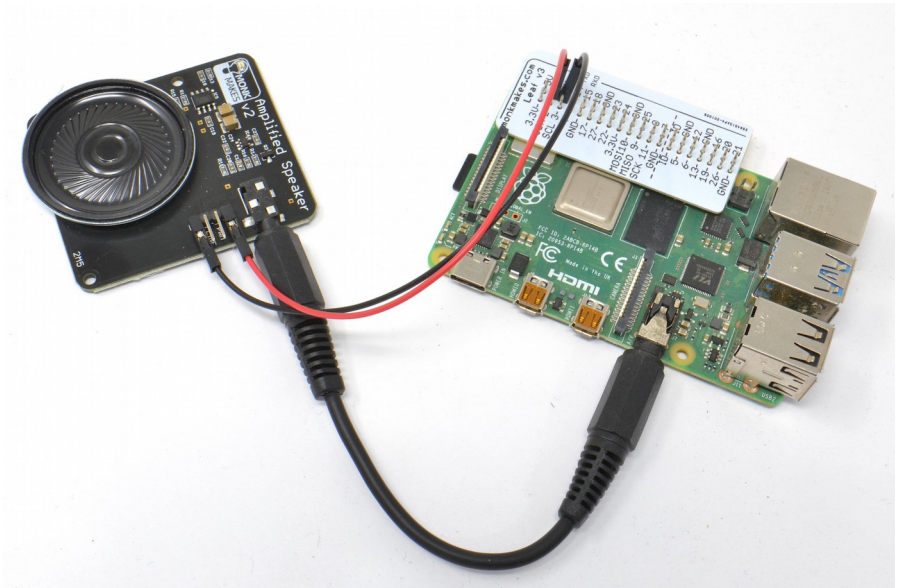
The MonkMakes Amplified Speaker 2 is an easy to use, low-power amplified speaker on a PCB.

The board needs a 3.3 to 6V power supply that can be provided by a Raspberry Pi, Raspberry Pi Pico, Arduino or any other microcontroller capable of supplying power at 300mA between that voltage range.

Audio input can use its the 3.5mm stereo audio jack socket (great for a Raspberry Pi 1 to 4). Stereo audio is mixed to a mono signal as there is just a single speaker on the board.

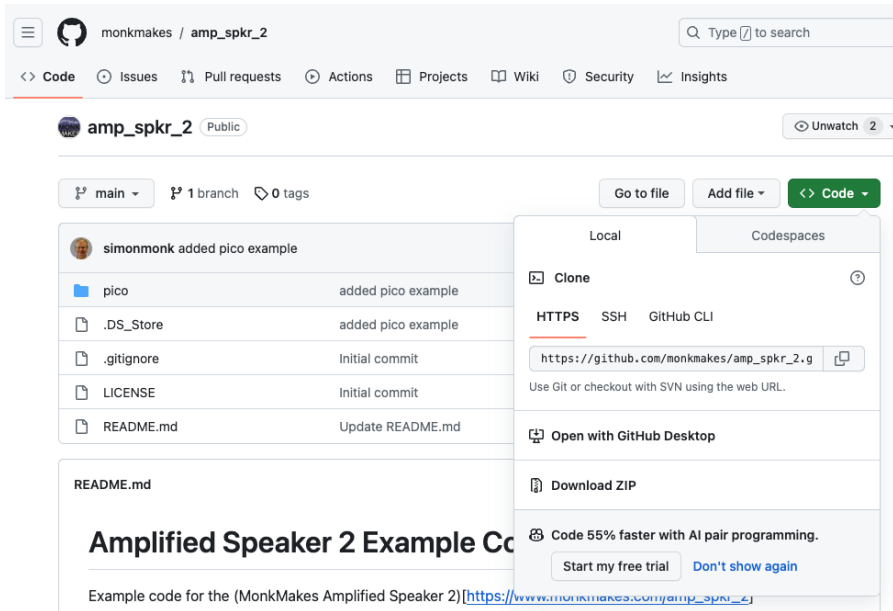
The board also has header pins that provide a line-in and low-pass filtered inputs, for use with PWM audio from microcontrollers.

The board can be powered from 3.3V to 6V with a maximum current consumption of 300mA.



# DOWNLOADING THE EXAMPLES

To download the ZIP archive containing example programs for all platforms, visit [https://github.com/monkmakes/amp\\_sprkr\\_2](https://github.com/monkmakes/amp_sprkr_2)



Click on the **Code** button, select Download ZIP and then extract the downloaded ZIP archive.

If you are familiar with git and would prefer to download the examples using the command line, then you can do so with the command:

```
$ git clone https://github.com/monkmakes/amp_sprkr_2
```

With the extracted archive, you will find folders containing example programs for:

- arduino – Arduino and other boards that can be programmed with the Arduino IDE
- esp32 – ESP32-based boards such as the ESP32 Lite and Devkit 1
- pico – MicroPython examples for the Raspberry Pi Pico and Pico W

# RASPBERRY PI 1-4

The MonkMakes Amplified Speaker 2 is a great way to add sound to your Raspberry Pi 1 to 4, using the Raspberry Pi's audio jack and female to female jumper wires to connect the Raspberry Pi to the Amplified Speaker.

## Connecting

Connect your Raspberry Pi to the MonkMakes Amplified Speaker 2 using female to female jumper wires as shown below. It's much easier to work out which pin is which if you use a GPIO template like the Raspberry Leaf (<https://monkmakes.com/leaf>).

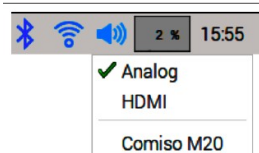
The connections are:

- GND to GND
- 5V on the Raspberry Pi to 3-6V on the Amplified Speaker 2

Then use a 3.5mm audio lead to connect the two audio sockets together.

## Selecting Audio Output

By default, if you have an HDMI monitor attached to your Raspberry Pi, audio will be routed through that and not the audio jack. To redirect it to the audio jack, select Analog from top right of the Raspberry Pi desktop.



## Testing Audio

There are many ways of playing sound on the Raspberry Pi. You can use VLC Player, or a web signal generator like this: <https://www.szynalski.com/tone-generator/>

## Raspberry Pi 5 and GPIO Pins

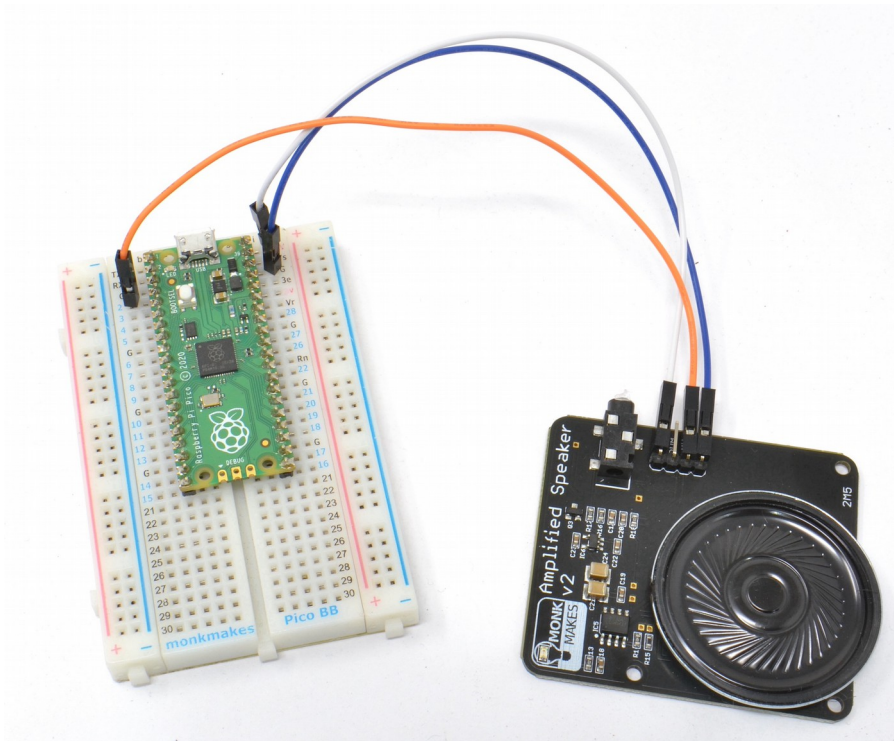
In theory, you should be able to use the GPIO pins of a Raspberry Pi 5, or one of the Zero models without an audio jack, but in practice, finding a reproducible way of redirecting audio to GPIO pins seems fraught with problems. If you get it to work, please let us know.

# RASPBERRY PI PICO AND PICO W

The connections are as follows for a Pico or Pico W:

- GND to GND
- 5V on the Pico to 3-6V on the Amplified Speaker 2
- Pin 2 on the Pico to Line on the Amplified Speaker 2.

You can use female-to-female jumper wires to connect the Amplified Speaker 2 directly to a Pico, or use a solderless breadboard and male to female jumper wires, as shown below.



Note that the breadboard shown above is customised to show the Pico's pins. For more information on this product see: [https://www.monkmake.com/pico\\_bb.html](https://www.monkmake.com/pico_bb.html)

## Simple Tone Generation

You can play simple tones (rough sounding square wave) using MicroPython's PWM feature. The frequency determines the tone and the duty cycle the volume. You will find an example of this in the file `simple_tone.py`.

```
from machine import Pin, PWM
from utime import sleep

speaker = PWM(Pin(2))

freq = 220
volume = 0.5 # 0.0 to 1.0

speaker.duty_u16(int(volume * 32767))
speaker.freq(freq)
sleep(1) # wait a second
speaker.duty_u16(0)
```

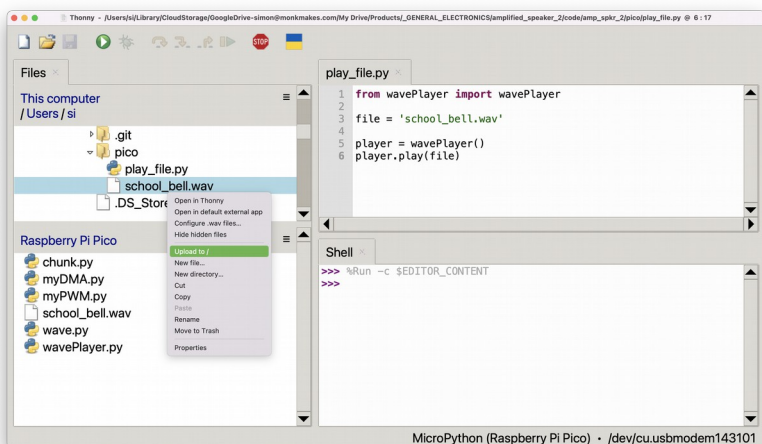
## Playing Samples

The instructions for playing a sound file are adapted from here:

<https://www.coderdojotc.org/micropython/sound/07-play-audio-file/>

It's a bit tricky to set up your Pico to do this, as it requires installing a few MicroPython libraries onto the Pico, that have to be installed by actually moving files, rather than using the Thonny.

When you have done this, the Pico will contain the files shown in the bottom left of the Figure below.



The process for putting files onto the Pico's file system is:

- Download the file onto your computer
- Navigate to the file using the 'This computer' section of Thonny
- Right-click on the file and select 'Upload to /'

Here's a list of the files and the URL from which the file can be downloaded:

- chunk.py - <https://raw.githubusercontent.com/joeky888/awesome-micropython-lib/master/Audio/chunk.py>
- wave.py - <https://raw.githubusercontent.com/joeky888/awesome-micropython-lib/master/Audio/wave.py>
- myDMA.py - <https://raw.githubusercontent.com/danjperron/PicoAudioPWM/main/myDMA.py>
- myPWM.py - <https://raw.githubusercontent.com/danjperron/PicoAudioPWM/main/myPWM.py>
- wavePlayer.py - <https://raw.githubusercontent.com/danjperron/PicoAudioPWM/main/wavePlayer.py>
- school\_bell.wav – From the pico folder of the examples download for this product (see Page 4).

Once the files are all copied onto the Pico, open the file *play\_file.py* from the product examples download (see Page 4) and run it. You should hear the sound of a bell ringing.

You can make your own WAV files using audio software such as Audacity. But you only have 2Mbytes of flash memory, that also has to contain all the libraries etc, so make them 8kHz and small!

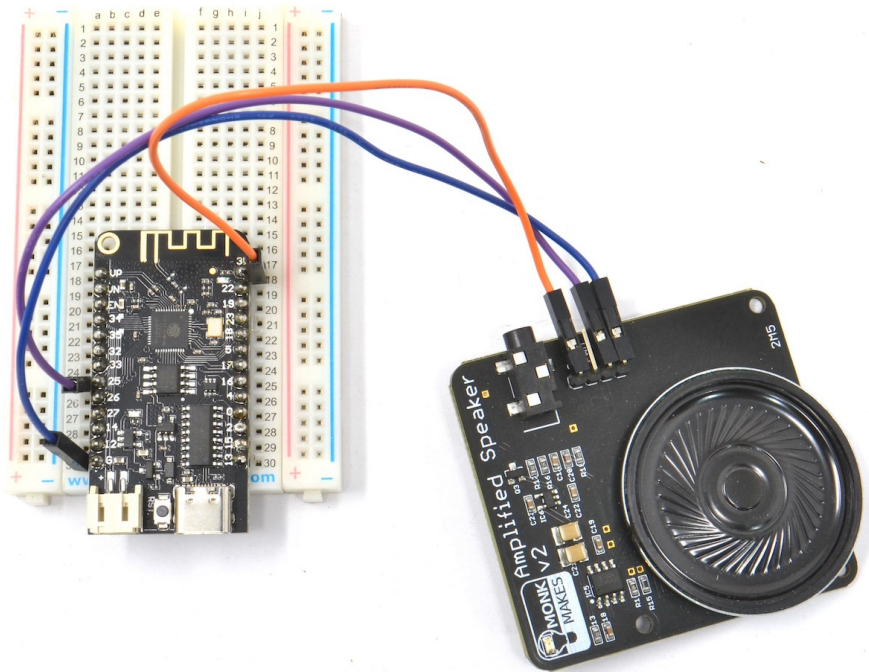


# ESP32

The connections are as follows for an ESP32 using the ADC channel on GPIO 25:

- GND to GND
- 3.3V on the ESP32 to 3-6V on the Amplified Speaker 2
- Pin 25 on the ESP32 to Line on the Amplified Speaker 2.

You can use female-to-female jumper wires to connect the Amplified Speaker 2 directly to a Pico, or use a solderless breadboard and male to female jumper wires, as shown below.



## Playing Samples

The following example code, which you will find in the download for the product in the file `esp32/play_file.py`, uses a hardware timer to play a raw 8-bit sound file at a sample rate of 8kHz. Only very short sound files of a few seconds can be played with this code.

The example sound `school_bell.raw` included in the downloads can be used, or you

can use Audacity to prepare a sound file. When exporting the file in Audacity make sure it has the following attributes as shown in the dialog below.

Export Audio

File Name:

school\_bell.raw

Folder:

/Users/si/Dropbox/Books in progress/53 ESP32/code/prog

Browse...

Format:

Other uncompressed files

Audio options

Channels

☒ Mono

☐ Stereo

☐ Custom mapping

Configure

Sample Rate

8000 Hz

Header

RAW (header-less)

Encoding

Unsigned 8-bit PCM

Export Range:

☒ Entire Project

☐ Multiple Files

☐ Current selection

☐ Trim blank space before first clip

Edit Metadata...

Cancel

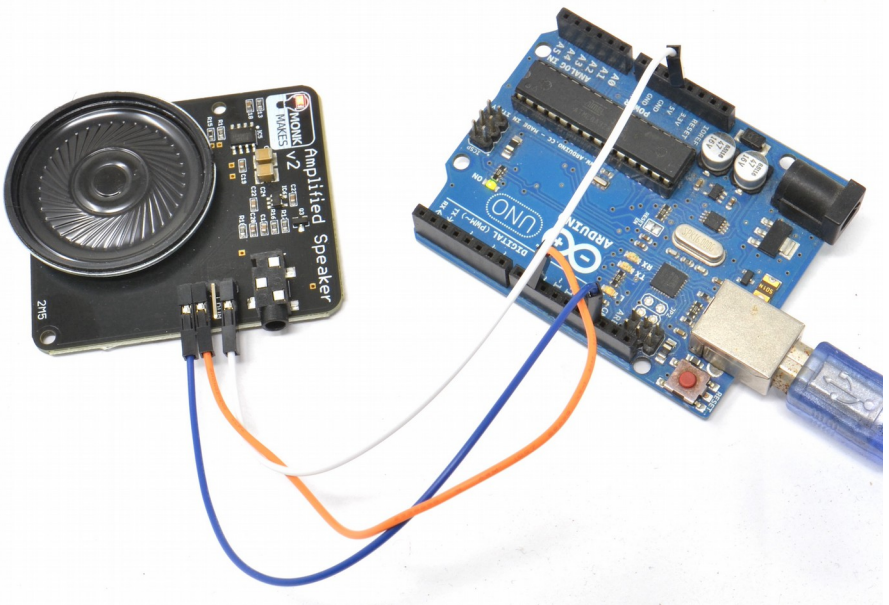
Export

# ARDUINO

In this section, we will use the popular Mozzi audio library for Raspberry Pi. Mozzi comes with a whole load of examples that you can load up and try out with your Amplified Speaker 2.

The connections are as follows:

- GND on the Arduino to GND on the Amplified Speaker 2
- 5V on the Arduino to 3-6V on the Amplified Speaker 2
- Pin 9 on the Arduino (or see Mozzi documentation) to PWM on the Amplified Speaker 2



## Simple Tones

You can play simple tones on the Arduino Uno using the `tone` command. You will find an example of this in the download for this product (see Page 4) inside the *arduino* folder. The sketch is called *simple\_scale*.

Upload the sketch and you should hear a rising scale repeating. The `tone` command will work on any pin of the Uno.

```

const int SPEAKER_PIN = 9;
const int notes[] = {131, 147, 165, 175, 196, 220, 247};

void setup() {}

void loop() {
  for (int i = 0; i < 7; i++) {
    tone(SPEAKER_PIN, notes[i]);
    delay(80);
  }
  delay(200);
}

```

## Samples and Wave Synthesis

Follow the instructions here for installing the Mozzi Arduino library and its examples.

<https://github.com/sensorium/Mozzi>

## Running Examples

In the Arduino IDE, select the File menu and then in the Examples submenu, find the Mozzi section at the bottom and pick an example to try.

Interesting examples to get started with are:

- FM Synth from 06 Synthesis
- PWM Phasing from 06 Synthesis
- Sample Table Arrays from 08 Samples
- Sample from 08 Samples

## TROUBLESHOOTING

**Problem:** The orange power LED in the MonkMakes logo does not light.

**Solution:** Check the jumper wires are connected correctly. Jumper wires can also fail, so try using different wires.

**Problem:** The orange power light fluctuates in brightness and the speaker sounds very distorted.

**Solution:** This is likely due to whatever is supplying power to the board is not capable of supplying sufficient current. The Speaker will draw a peak current of 300mA.

**Problem:** There is a lot of hiss, or hum, even when no sound is playing.

**Solution:** Check that the jumper wires are not close to any sources of electrical noise, like AC power leads. Also keeping the jumper wires short may help.

If using jumper wires for the audion connection, try both input pins (PWM and Line), one may be better than the other.

**Problem:** The sound is a bit 'tinny' and not very loud.

**Solution:** This is a small low power speaker. You are never going to get good sound quality out of it.

## SUPPORT

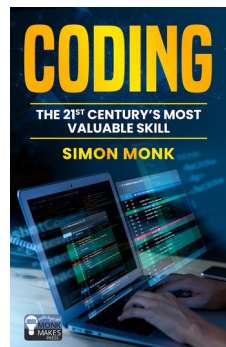
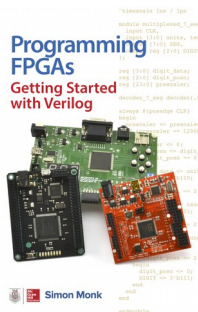
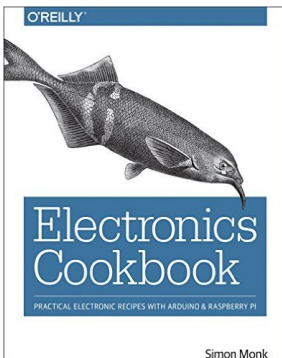
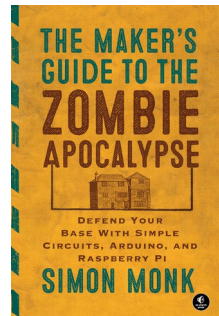
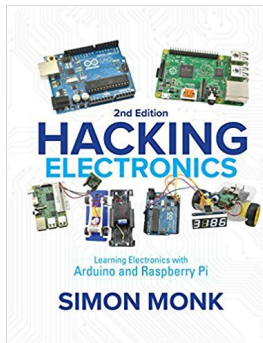
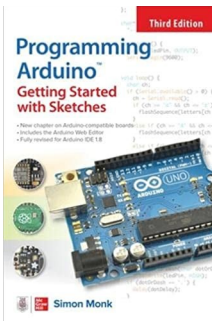
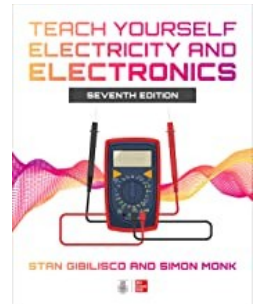
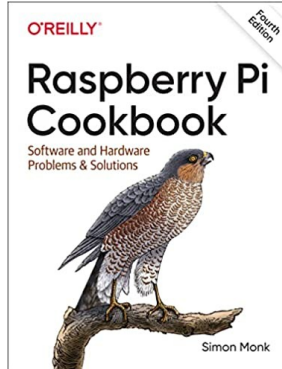
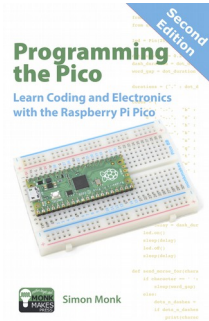
You can find the Product's information page here:

[https://monkmakes.com/amp\\_spkr\\_2](https://monkmakes.com/amp_spkr_2) including a datasheet for the product.

If you need further support, please email [support@monkmakes.com](mailto:support@monkmakes.com).

# Books

The designer of this product (Simon Monk) also writes books about electronics. You may find some of these interesting.



# MONKMAKES

As well as this product, MonkMakes makes all sorts of kits and gadgets to help with your electronics projects. Find out more, as well as where to buy here:

<https://www.monkmakes.com/products> you can also follow MonkMakes on Twitter @monkmakes.

