Adafruit I2S Amplifier BFF
Created by Kattni Rembor

https://learn.adafruit.com/i2s-amplifier-bff

Last updated on 2023-08-29 04:56:59 PM EDT
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overview</td>
<td>3</td>
</tr>
<tr>
<td>Pinouts</td>
<td>6</td>
</tr>
<tr>
<td>• I2S Amplifier</td>
<td></td>
</tr>
<tr>
<td>• Speaker Connector</td>
<td></td>
</tr>
<tr>
<td>• I2S Pins</td>
<td></td>
</tr>
<tr>
<td>• I2S Default Pin Jumpers</td>
<td></td>
</tr>
<tr>
<td>• I2S Through-Hole Pads</td>
<td></td>
</tr>
<tr>
<td>• Left/Right Channel Jumper</td>
<td></td>
</tr>
<tr>
<td>CircuitPython</td>
<td>9</td>
</tr>
<tr>
<td>• Wiring</td>
<td></td>
</tr>
<tr>
<td>• CircuitPython Usage</td>
<td></td>
</tr>
<tr>
<td>CircuitPython Docs</td>
<td>13</td>
</tr>
<tr>
<td>Arduino</td>
<td>13</td>
</tr>
<tr>
<td>• Wiring</td>
<td></td>
</tr>
<tr>
<td>• Tone Example Code</td>
<td></td>
</tr>
<tr>
<td>• Audio Playback Example Code</td>
<td></td>
</tr>
<tr>
<td>Downloads</td>
<td>18</td>
</tr>
<tr>
<td>• Files</td>
<td></td>
</tr>
<tr>
<td>• Schematic and Fab Print</td>
<td></td>
</tr>
</tbody>
</table>
Our QT Py boards are a great way to make very small microcontroller projects that pack a ton of power - and now we have a way for you to add an I2S 3 Watt amplifier, for high quality audio playback, that can fit on the back of your miniature dev board. It uses just three GPIO pins that do not intersect with the I2C/UART or SPI port.

We call this the Adafruit I2S Amp BFF - a "Best Friend Forever". When you were a kid you may have learned about the "buddy" system, well this product is kinda like that! A board that will watch your QT Py's back and give it more capabilities.
This PCB is designed to fit onto the back of any QT Py or Xiao board, it can be soldered into place or use pin and socket headers to make it removable. Onboard is a MAX98357 audio amplifier and picoblade-compatible connector for plugging in a 4 or 8 ohm speaker. We use A0 for the audio data, A1 for wordselect clock, and A2 for bitclock. This pinout will work with ESP32 series, nRF52840 and RP2040 chipset boards. It won't work with the ATSAMD21 'original 'QT Py because those pins on the SAMD21 are not I2S capable (). However, you could cut and rewire the traces to connect to the I2S pads if desired - personally we recommend just upgrading to an RP2040 QT Py instead.
We include some header that you can solder to your QT Py. You can also pick up an Itsy Bitsy short female header kit to make it removable but compact, you'll just need to trim down the headers to 7 pins long.

- Comes as an assembled and tested PCB
- For any QT Py or Xiao boards
- Contains a MAX98357 3 Watt audio amplifier pre-configured for 'stereo' mix output and 9 dB gain which will work great for any project.
- Connect to the speaker output with a picoblade-compatible 2-pin cable. We recommend this 3W 4 ohm speaker.
- There are various Arduino / CircuitPython / MicroPython libraries can be used to talk to the I2S amplifier.

Speaker and QT Py are not included.
Pinouts

I2S Amplifier

- The grey square located in the center at the top of the BFF is the MAX93785 I2S amplifier chip. It supports digital audio.

Speaker Connector

- This two-pin PicoBlade-compatible speaker connector, located at the center of the BFF, is meant for plugging in a 4-8Ω, 3W-or-less speaker.
- To connect a speaker with bare wires, you'll want to pick up this cable.
- If you need a speaker, this speaker is an excellent place to start.

I2S Pins

These pins are compatible with ESP32-series, nRF52840 and RP2040 microcontroller boards.

- A0 - This is used for audio data, or DIN.
- A1 - This is used for wordselect clock, or LRC.
- A2 - This is used for bitclock, or BIT.

These pins will not work with the QT Py SAMD21 microcontroller board. To use a SAMD21, you would need to cut the pin jumpers and rewire the traces to compatible pins.
I2S Default Pin Jumpers

There are jumpers on the I2S pins to enable you to change the default QT Py pins.

First, cut the trace on the jumper associated with the default pin. Next, solder a wire between the jumper pad furthest away from the pin label on the silk and the desired QT Py pin.

- A0 jumper - Cutting this trace enables you to reassign the DIN (audio data) to a different pin.
- A1 jumper - Cutting this trace enables you to reassign the LRC (wordselect clock) to a different pin.
- A2 jumper - Cutting this trace enables you to reassign the BIT (bitclock) to a different pin.

The pads highlighted in magenta in the image below are the correct pads to wire to a new pin.

I2S Through-Hole Pads

You can use these through-hole pads to connect the I2S pins to different QT Py pins. They are labeled on the back of the board.
To reassign an I2S pin, cut the trace on the jumper associated with the default pin, and then solder a wire between the through-hole pad and the desired QT Py pin.

- DIN - This pad, located next to the A0 pin label on the board silk, is connected to the DIN pin.
- LRC - This pad, located between the DIN pin and the MAX93785 chip, is connected to the LRC pin.
- BIT - This pad, located near the A3 pin label on the board silk, is connected to the BIT pin.

Left/Right Channel Jumper

This jumper allows you to choose between the left channel, right channel, or both together.

- The BFF defaults to stereo mix, using both the left and right channel equally if no changes are made to this jumper.
- If you solder the pad near the R label on the board silk and the center pad together, you will get audio out of the right channel only.
- If you solder the pad near the L label on the board silk and the center pad together, you will get audio out of the left channel only.
CircuitPython supports sending I2S audio signals using the builtin `audiobusio` module, making it simple to use the I2S BFF with your QT Py. It's easy to use CircuitPython to play different types of audio using I2S, including tones and WAV files.

This page uses a QT Py RP2040 as the microcontroller board. These demos will work with any QT Py version, as long as it has the `audiobusio` module builtin to CircuitPython for that board. Visit circuitpython.org/downloads and search for the QT Py you wish to use. Click on it, and you'll find a list of built-in modules. The list for the QT Py RP2040 is indicated by the magenta rectangle. `audiobusio` is highlighted.

**Wiring**

You'll need to connect the BFF to a QT Py RP2040, and a speaker to the BFF.
QT Py to BFF

Connect a QT Py RP2040 to the I2S BFF by soldering on a pair of pin headers to one, and a pair of socket headers to the other.

In this setup, the pin headers are on the QT Py on the top of the image, and the socket headers are on the I2S BFF on the bottom of the image. They should be plugged in with the backs of the boards facing each other. The I2S BFF has on the back, a USB label with an arrow, to indicate which end of the board should be aligned with the USB connector on the QT Py.

If your BFF is not working as expected, verify that you have attached them in the right orientation with the USB indicator on the BFF on the same end as the USB connector on the QT Py.

I2S BFF to Speaker

Connect a speaker to your I2S BFF as shown below.

Plug the PicoBlade-compatible connector on your speaker into the PicoBlade-compatible socket on your I2S BFF.

The socket is keyed along with the connector to ensure avoiding plugging it in backwards.

Speaker to Connector Cable (Optional)

If your speaker comes with bare wires, you'll want to pick up this cable, and solder it to your speaker.
Many speakers come with bare wires attached. To use those speakers with the I2S BFF, you'll need to purchase a bare-wires-to-PicoBlade-compatible-connector wire, and solder the wires to the same color wires on your speaker.

Solder the cable to the speaker by connecting the wires on the cable to the same color wires on the speaker.

CircuitPython Usage

There are two I2S audio examples: tone playback and WAV playback.

Tone Playback

Click the Download Project Bundle button below to download the code.py file in a zip file. Extract the contents of the zip, and copy the code.py file to your CIRCUITPY drive.

```python
# SPDX-FileCopyrightText: 2023 Kattni Rembor for Adafruit Industries
# SPDX-License-Identifier: MIT

"""
CircuitPython I2S Tone playback example.
"""
import time
import array
import math
import audiocore
import board
import audiobusio

TONE_VOLUME = 0.1  # Increase this to increase the volume of the tone.
FREQUENCY = 440    # Set this to the Hz of the tone you want to generate.

audio = audiobusio.I2SOut(board.A2, board.A1, board.A0)

length = 8000 // FREQUENCY
sine_wave = array.array("h", [0] * length)
for i in range(length):
    sine_wave[i] = int((math.sin(math.pi * 2 * i / length)) * TONE_VOLUME * (2 ** 15 - 1))
sine_wave_sample = audiocore.RawSample(sine_wave)

while True:
    audio.play(sine_wave_sample, loop=True)
    time.sleep(1)
```

© Adafruit Industries
Once successfully copied, you'll begin hearing a one second 440Hz tone, every other second!

You can edit the following variables in the example code to increase the tone volume, and change the frequency in Hz of the tone being played.

```
TONE_VOLUME = 0.1  # Increase this to increase the volume of the tone.
FREQUENCY = 440    # Set this to the Hz of the tone you want to generate.
```

WAV Playback

Click the Download Project Bundle button below to download the code.py file and the booploop.wav file in a zip file. Extract the contents of the zip, and copy the code.py and booploop.wav files to your CIRCUITPY drive.

Your CIRCUITPY drive contents should resemble the following.

```
# SPDX-FileCopyrightText: 2023 Kattni Rembor for Adafruit Industries
# SPDX-License-Identifier: MIT

# CircuitPython I2S WAV file playback.
import audiocore
import board
import audiobusio

LOOP = False  # Update to True loop WAV playback. False plays once.

audio = audiobusio.I2SOut(board.A2, board.A1, board.A0)

with open("chikken.wav", "rb") as wave_file:
    wav = audiocore.WaveFile(wave_file)

print("Playing wav file!")
audio.play(wav, loop=LOOP)
while audio.playing:
    pass

print("Done!")
```

Once successfully copied, you'll hear the WAV file play once.
You can update the `LOOP` variable in this example to `True` if you want the WAV playback to loop.

```python
LOOP = False  # Update to True loop WAV playback. False plays once.
```

---

### CircuitPython Docs

[CircuitPython Docs ()](https://docs.circuitpython.org/en/latest/)

---

### Arduino

Using the I2S BFF and the QT Py RP2040 with Arduino involves soldering up the two boards, connecting them in the appropriate orientation, and running the provided example code. These examples do not require any separate libraries.

#### Wiring

You'll need to connect the BFF to a QT Py RP2040, and a speaker to the BFF.

#### QT Py to BFF

Connect a QT Py RP2040 to the I2S BFF by soldering on a pair of pin headers to one, and a pair of socket headers to the other.

In this setup, the pin headers are on the QT Py on the top of the image, and the socket headers are on the I2S BFF on the bottom of the image. They should be plugged in with the backs of the boards facing each other. The I2S BFF has on the back, a USB label with an arrow, to indicate which end of the board should be aligned with the USB connector on the QT Py.

If your BFF is not working as expected, verify that you have attached them in the right orientation with the USB indicator on the BFF on the same end as the USB connector on the QT Py.
I2S BFF to Speaker

Connect a speaker to your I2S BFF as shown below.

Plug the PicoBlade-compatible connector on your speaker into the PicoBlade-compatible socket on your I2S BFF.

The socket is keyed along with the connector to ensure avoiding plugging it in backwards.

Speaker to Connector Cable (Optional)

If your speaker comes with bare wires, you'll want to pick up [this cable](#), and solder it to your speaker.

Many speakers come with bare wires attached. To use those speakers with the I2S BFF, you'll need to purchase a bare-wires-to-PicoBlade-compatible-connector wire, and solder the wires to the same color wires on your speaker.
Solder the cable to the speaker by connecting the wires on the cable to the same color wires on the speaker.

Tone Example Code

This example plays a tone. The code below was adapted from the SimpleTone.ino demo in the arduino-pico core I2S library examples.

```c
#include <I2S.h>

// Create the I2S port using a PIO state machine
I2S i2s(OUTPUT);

// GPIO pin numbers
#define pBCLK A2 // QT Py BFF default BITCLOCK
#define pWS   A1 // QT Py BFF default LRCLOCK
#define pDOUT A0 // QT Py BFF default DATA

const int frequency = 440; // frequency of square wave in Hz
const int amplitude = 500; // amplitude of square wave
const int sampleRate = 16000; // 16 KHz is a good quality
const int halfWavelength = (sampleRate / frequency); // half wavelength of square wave
```

©Adafruit Industries

Page 15 of 19
int16_t sample = amplitude;  // current sample value
int count = 0;

void setup() {
  Serial.begin(115200);
  while (!Serial) delay(10);
  Serial.println("I2S simple tone");

  i2s.setBCLK(pBCLK);
  i2s.setDATA(pDOUT);
  i2s.setBitsPerSample(16);

  // start I2S at the sample rate with 16-bits per sample
  if (!i2s.begin(sampleRate)) {
    Serial.println("Failed to initialize I2S!");
    while (1);  // do nothing
  }
}

void loop() {
  if (count % halfWavelength == 0) {
    // invert the sample every half wavelength count multiple to generate square
    // wave
    sample = -1 * sample;
  }

  // write the same sample twice, once for left and once for the right channel
  i2s.write(sample);
  i2s.write(sample);

  // increment the counter for the next sample
  count++;
}

Once you have uploaded the sketch to your QT Py, you’ll hear a tone.

You can change the frequency (pitch) of the tone by updating the frequency variable.

const int frequency = 440;  // frequency of square wave in Hz

Audio Playback Example Code

This example plays a PCM audio file when the Boot button is pressed. Check out this Python script to convert WAV files to PCM files.

Click the button below to download the source code and header file. Unzip it, and open it with the Arduino IDE.

Audio Playback Source Code

When you open the code in the Arduino IDE, you will see the Arduino sketch code in one tab, and the header file in a second tab.
```c
#include <I2S.h>
#include "startup.h" // audio file in flash

// Create the I2S port using a PIO state machine
I2S i2s(OUTPUT);

// GPIO pin numbers
#define pBCLK A2 // QT Py BFF default BITCLOCK
#define pWS   A1 // QT Py BFF default LRCLOCK
#define pDOUT A0 // QT Py BFF default DATA
#define USERBUTTON 21 // QT Py RP2040 built in button

// variable shared between cores
volatile bool playaudio = false;

void setup() {
  Serial.begin(115200);
  //while (!Serial) delay(10);
  Serial.println("I2S playback demo");
  pinMode(USERBUTTON, INPUT_PULLUP);
}

void loop() {
  // on button press tell the other core to play audio clip!
  if (!digitalRead(USERBUTTON)) {
    playaudio = true;
  } else {
    playaudio = false;
  }
}

void setup1() {
  i2s.setBCLK(pBCLK);
  i2s.setData(pDOUT);
  i2s.setBitsPerSample(16);
}

void loop1() {
  // the main loop will tell us when it wants us to play!
  if (playaudio) {
    play_i2s(startupAudioData, sizeof(startupAudioData), startupSampleRate);
  }
}

void play_i2s(const uint8_t *data, uint32_t len, uint32_t rate) {
```
```cpp
// start I2S at the sample rate with 16-bits per sample
if (!i2s.begin(rate)) {
    Serial.println("Failed to initialize I2S!");
    delay(500);
    i2s.end();
    return;
}

for(uint32_t i=0; i<len; i++) {
    uint16_t sample = (uint16_t)data[i] << 6; // our data is 10 bit but we want 16 bit so we add some gain
    // write the same sample twice, once for left and once for the right channel
    i2s.write(sample);
    i2s.write(sample);
}

i2s.end();
}
```

Once you've uploaded the sketch onto the QT Py RP2040, press the Boot button to hear the clip played.

The Boot button (highlighted in blue in the image), is located towards the bottom edge of the board between the STEMMA QT connector on the right and the USB connector on the left.

---

**Downloads**

**Files**

- [MAX98357 Datasheet ()](#)
- [EagleCAD PCB files on GitHub ()](#)
- [Fritzing object in the Adafruit Fritzing Library ()](#)
Schematic and Fab Print