Audio Codec Breakout - WM8960 Hookup Guide

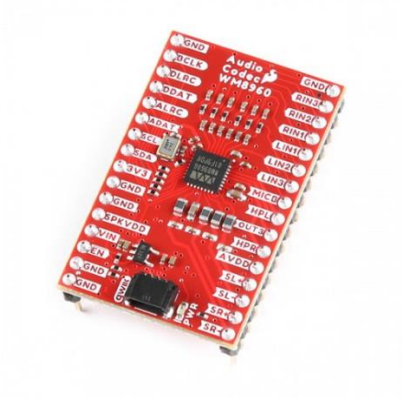
Introduction

https://youtu.be/NvHIXSyEeDA

The SparkFun Audio Codec Breakout - WM8960 is a low power, high quality stereo codec with 1W Stereo Class D speaker drivers and headphone drivers. The WM8960 acts as a stereo audio ADC and DAC, and communicates using I^2S, a standard audio data protocol (not to be confused with I^2C). This audio codec is chock full of features some of which includes advanced on-chip digital signal processing for automatic level control (ALC) for the line or microphone input, programmable gain amplifier (PGA), pop and click suppression, and its ability to configure I^2S settings and analog audio path through software via I^2C.

SparkFun Audio Codec Breakout - WM8960 (Qwiic)

BOB-21250 $17.95



SparkFun Audio Codec Breakout - WM8960 with Headers (Qwiic)

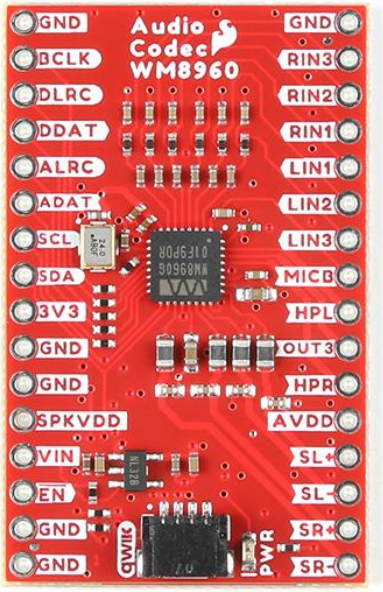
BOB-21772 $18.50

Required Materials

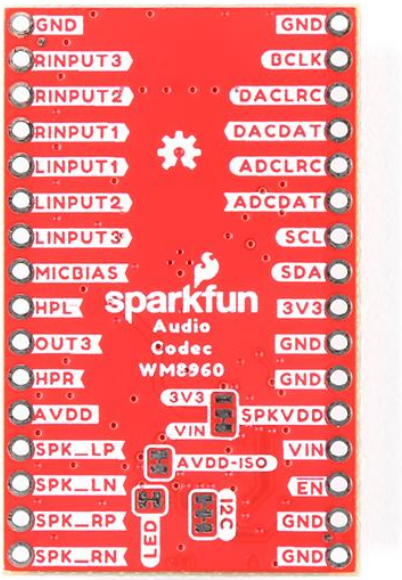
To follow along with this tutorial, you will need the following materials at a minimum. You may not need everything though depending on what you have. Add it to your cart, read through the guide, and adjust the cart as necessary. Note that the wishlist does not include any microphones or speakers.

Hardware Overview

We broke out the pins of the WM8960 onto a breakout board to standard 0.1" spaced PTH. We'll highlight relevant application circuits, hardware, and pins that are broken out in this section.

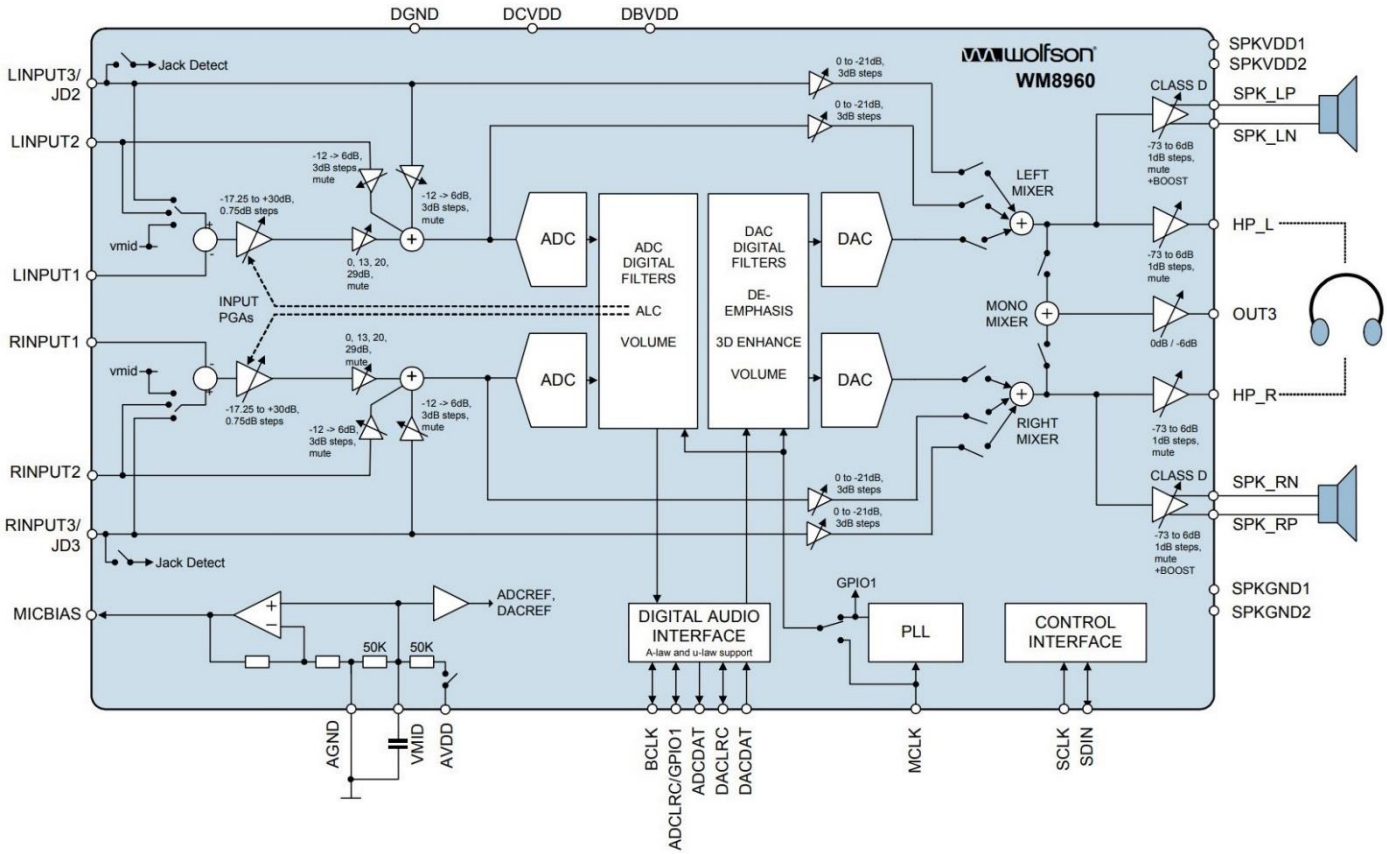


Top View



Bottom View

Keep in mind that the WM8960 is chock full of features as you can see from the datasheet's block diagram below. The input and output will depend on your project's needs. We recommend referencing the block diagrams listed in the datasheet to follow along. For more information, check out the datasheet linked in the Resources and Going Further.

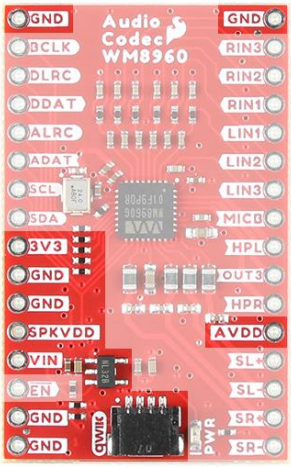


Block diagram from the WM8960 Datasheet (v4.2) on page 1.

Power

There are a variety of power and power-related nets broken out to the Qwiic connector and through hole pads on the edge of the board. The nets consist of the following.

- **VIN** - Voltage input for the analog and speaker circuits. The input voltage is between **3.7V** and **5.5V**.
    - **SPKVDD** - By default, the **VIN** pin is connected directly to **SPKVDD**. The connection can be disabled but cutting the jumper on the back between **VIN** and **SPKVDD**. There is an option to connect the **SPKVDD** pad to the **3V3** pin by adding a solder blob between the two pads. If you decide to provide your own voltage, users will need to keep the jumpers open. The recommended voltage for this pin is between **2.7V** and **5.5V**.
    - **AVDD** - Voltage from the **VIN** pin is regulated down to **3.3V** from the XC6222 3.3V/700mA voltage regulator. This voltage is used to power the analog circuit. For users that want to provide their own voltage, cut the jumper on the back of the board labeled as **AVDD-ISO**. The recommended voltage for this pin is between **2.7V** and **3.6V**.
- **3V3** - This pin is broken out on the edge of the board as well as the Qwiic connector. This voltages is used to power the digital circuit. This voltage is tied to the IC's DCVDD and DBVDD pins. Note that this is on a separate net and not connected to the 3.3V voltage regulator. The recommended voltage for this pin is between **1.7V** and **3.6V**. This will typically be **3.3V** when connecting to a microcontroller's 3.3V voltage regulator and Qwiic connector.
- **GND** - Common, ground voltage (0V reference) for the system.



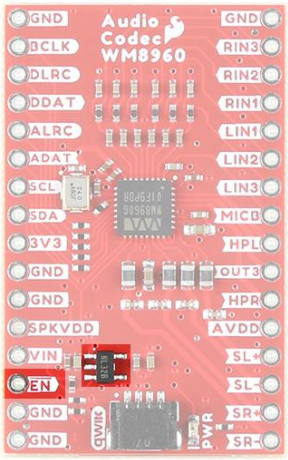
To get started powering the audio codec, you will just need to provide power to the **VIN** and **3V3** pins. We recommend using your development board's voltages without adjusting the jumpers. For example, the IoT RedBoard - ESP32 Development Board can provide 5V through the female header and 3.3V through the Qwiic connector. Users can connect 5V to the audio codec's VIN pin to power the analog and speaker circuit. To power the digital circuit, users can provide 3.3V by adding a Qwiic cable between the Qwiic connectors. Of course, you will need to also connect your reference voltage for your system. By using the Qwiic cable, you will be connecting GND on both boards.

⚡ **Warning!** If you decide to cut the trace between **VDD** and **SPKVDD** to connect **SPKVDD** to **3V3**, make sure to avoid any shorts between **VDD** and **SPKVDD**. If you place an input voltage on **VIN** that is higher than **3V3** pin is rated for will damage the WM8960 and anything connected to **3V3**.

For users providing their own power supply for **SPKVDD** and **AVDD**, make sure to keep the jumpers open.

Enable Pin

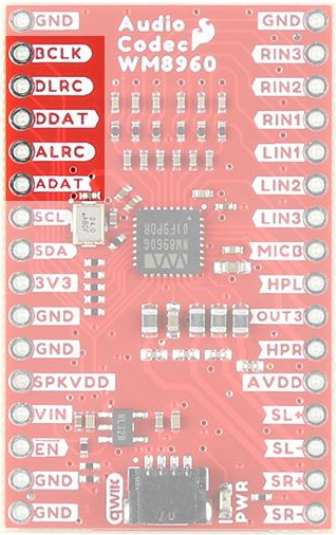
The enable pin (labeled as EN) on the edge of the board is active low. Pulling the pin low will disable the 3.3V voltage regulator XC6222 that is connected to AVDD.



Digital Audio Interface

The board breaks out pins for the digital audio interface. This is used for inputting DAC data into the WM8960 and outputting ADC data from the IC.

- **BCLK** - Bit clock, for synchronization. This can be set as an input or output depending on the configuration.
- **DLRC** or **DACLRC** - DAC data left and right alignment clock. This can be set as an input or output depending on the configuration.
- **DDAT** or **DACDAT** - DAC data input.
- **ALRC** or **ADCLRC** - ADC data left and right alignment clock. This can be set as an input or output depending on the configuration.
    - This pin can also be configured as a GPIO pin (e.g. GPIO1). As stated in the datasheet, the ADC will use the DACLRC as a frame clock for the ADCs and DACs. The ADCLRC/GPIO1 pin function should not be modified when the ADC is enabled. For more information check out the section about the Digital Audio Interface in the datasheet on pg 48.
- **ADAT** or **ADCDAT** - ADC data output.

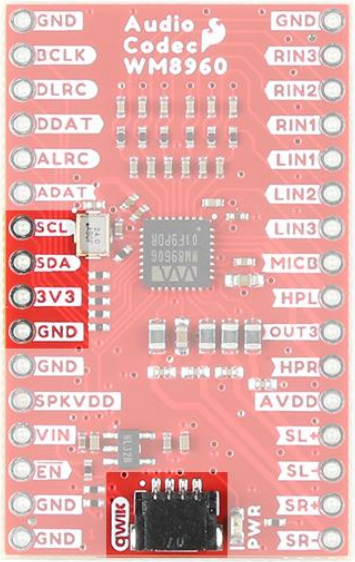


Four audio data formats listed below are supported.

- Left justified
- Right justified
- I^2S (not to be confused with I^2C)
- DSP

Qwiic and I²C

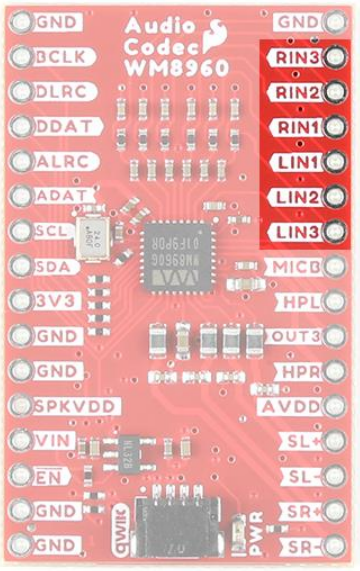
The board includes one Qwiic connector to configure the I²S settings and analog audio path. For users that need to solder directly to the board, the pins are also broken out on the edge PTH. The I²C data and clock lines are also tied to 2.2kΩ pull-up resistors. The default address of the WM8960 is **0x1A**. *Note that the datasheet shows the address as 0x34h, which is 0x1A shifted left 1 bit and then includes the appended "0" (aka write bit).*



Stereo Input

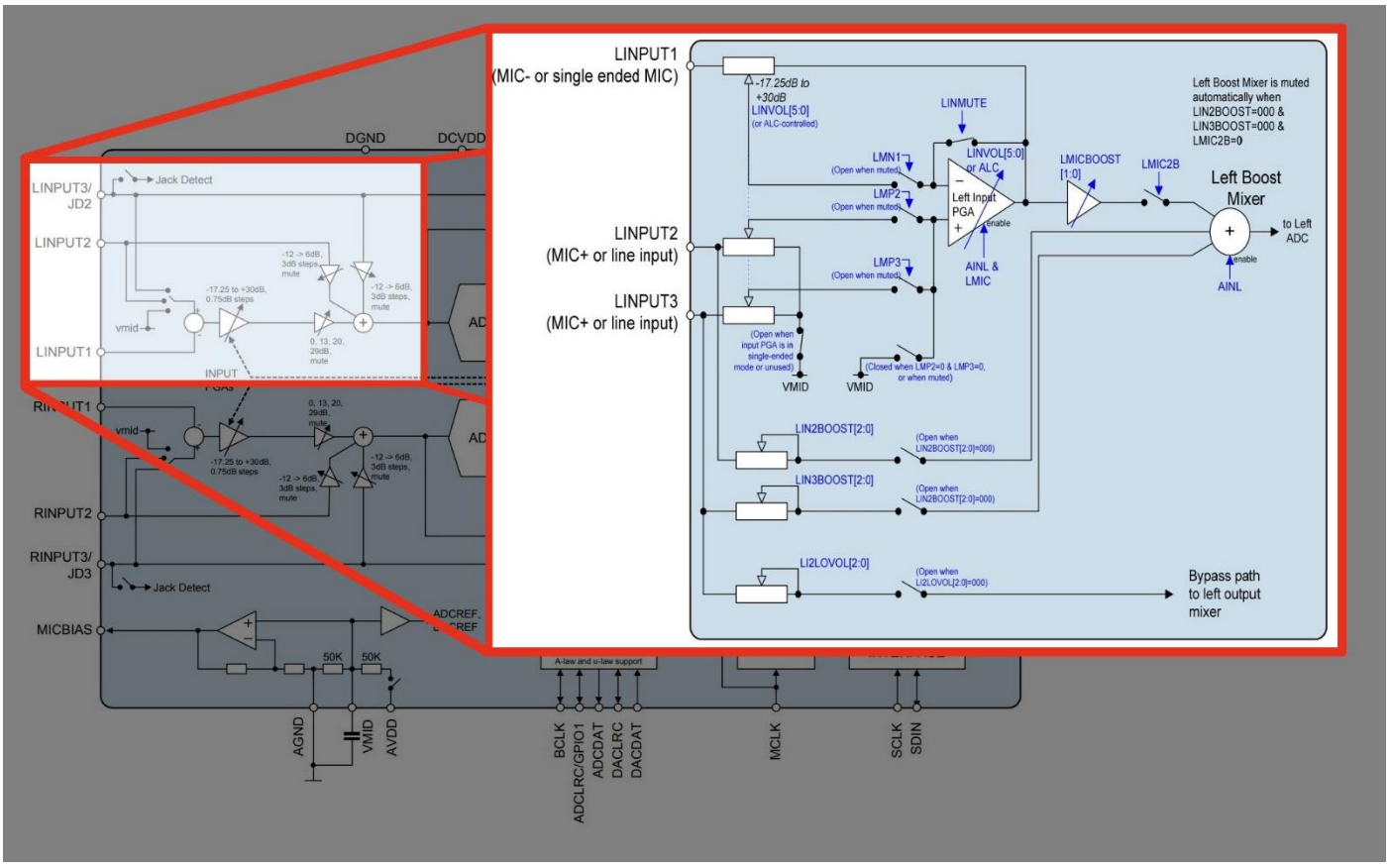
The WM8960 includes 6x flexible analog input pins. These pins can be used for line level or microphone level (balanced or un-balanced). Note that balanced microphones refers to differential microphones while un-balanced microphones refers to single-ended microphones.

- **RIN3** / **RINPUT3** - Right analog input 3 pin. This is connected to *RINPUT3* or JD3.
  - right channel, line input
  - right channel, differential +MIC input
  - jack detect input pin
- **RIN2** / **RINPUT2** - Right analog input 2 pin. This is connected to *RINPUT2*.
  - right channel, line input
  - right channel, differential +MIC input
- **RIN1** / **RINPUT1** - Right analog input 1 pin. This is connected to *RINPUT1*.
  - right channel, single-ended MIC input
  - right channel, differential -MIC input
- **LIN1** / **LINPUT1** - Left analog input 1 pin. This is connected to *LINPUT1*.
  - left channel, single-ended MIC input
  - left channel differential -MIC input
- **LIN2** / **LINPUT2** - Left analog input 2 pin. This is connected to *LINPUT2*.
  - left channel, line input
  - left channel, differential +MIC input
- **LIN3** / **LINPUT3** - Left analog input 3 pin. This is connected to *LINPUT3* or *JD2*.
  - left channel, line input
  - left channel, differential +MIC input
  - jack detect input pin



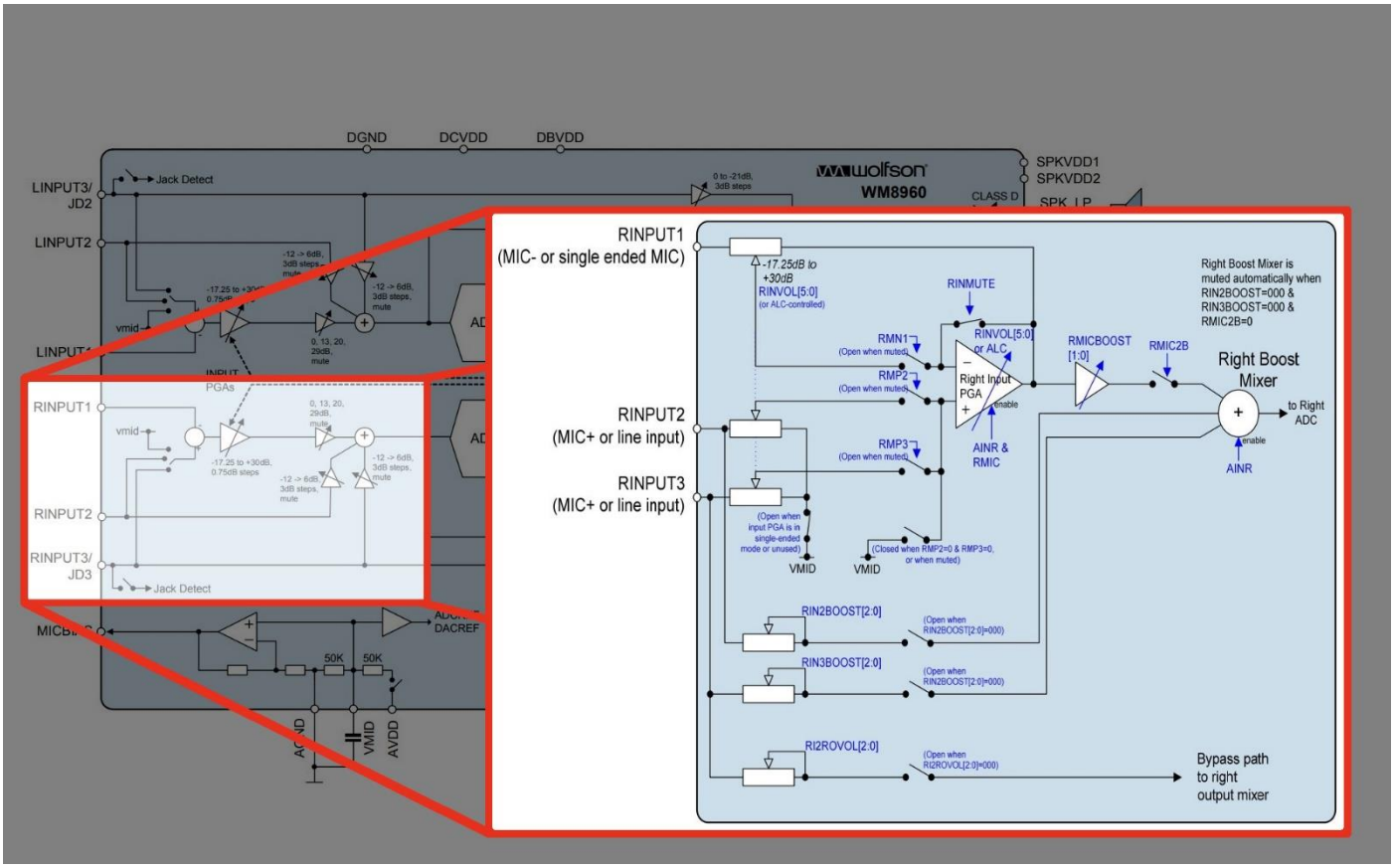
Remember the block diagram that was linked earlier just before soldering the headers to the board? The datasheet goes into more detail with the input path for the left and right channel.

Below is a modified image of the block diagram that highlights the input signal path before the left ADC for the left input channels on page 19.



Left Input Signal Path from the WM8960 Datasheet (v4.2) on page 19.
Note: Note that the pins are flipped from the initial block diagram and the left input channels. Instead of LINPUT3 at the top, it starts with LINPUT1.

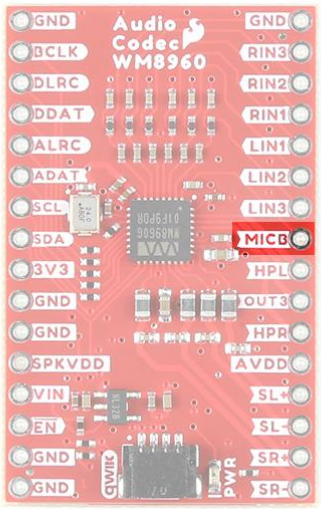
Below is a modified image of the block diagram that highlights the input signal path before the right ADC for the right input channels on page 20.



Right Input Signal Path from the WM8960 Datasheet (v4.2) on page 20.

Microphone Bias

The **MICB** or **MICBIAS** provides a low noise reference voltage for biasing eletret condenser microphones (ECMs).

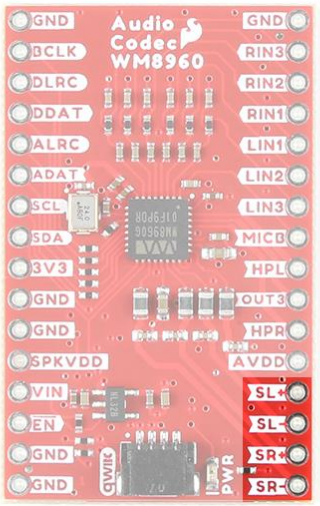


Speaker Out

The WM8960 includes differential stereo speaker outputs for the left and right speaker. The speaker is rated as a class D speaker driver and can drive 1W into 8Ω speakers. For users looking to power the speakers with a separate power supply, you can cut the jumper between the pads labeled as **SPKVDD** and **VIN** on the back of the board.

- **SL+** / **SPK_LP** - Speaker left positive output.
- **SL-** / **SPK_LN** - Speaker left negative output.

- **SR+** / **SPK_RP** - Speaker right positive output.
- **SR-** / **SPK_RN** - Speaker right negative output.

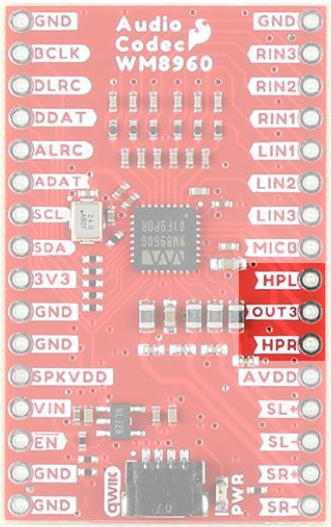


Note: When wiring differential speakers to speaker out terminals, make sure to match the "+" to "+" and "-" to "-" for both speakers. If one of the stereo speakers is wired "+" to "-" and "-" to +" while the other speaker is wired "+" to "+" and "-" to "-", the audio signals will interfere and cancel each other out. If there are no labels on the speakers, just make sure to wire the speakers consistently.

Headphones Output

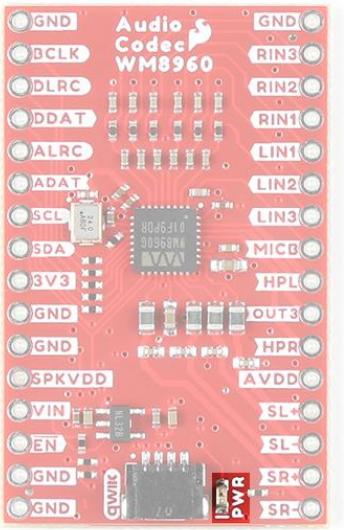
The WM8960 can drive 16Ω and 32Ω headphones. The connection will depend on your setup. Make sure to check out the **Headphone Output** in the datasheet on page 41 for more information.

- **HPL** - The left headphone out will be connected to the Tip of a TRS connector.
- **OUT3** - This is connected to the Sleeve of a TRS connector. If using DC blocking capacitors, connect this pin to AGND.
- **HPR** - The right headphone out will be connected to the Ring of a TRS connector.



LED

The board includes one LED labeled as **PWR**. This indicates when there is power on the 3.3V net to power the IC's digital circuit. You can disable it with the jumper on the back of the board.

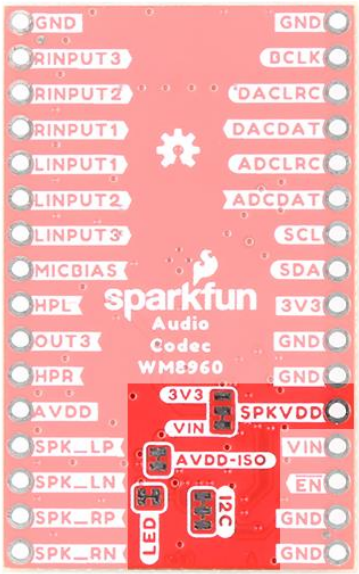


Jumpers

Note: If this is your first time working with jumpers, check out the How to Work with Jumper Pads and PCB Traces tutorial for more information.

The following ten jumpers are included on the Audio Codec Breakout - WM8960.

- **VIN**, **SPKVDD**, and **3V3** - This 3-pad jumper allows users to select a voltage source for the speaker VDD (**SPKVDD**). By default, the center pad (**SPKVDD**) is closed on the **VIN** pad. Cut this trace and add a solder blob on the **3V3** side to provide **SPKVDD** with 3.3V. Leave both jumpers open to provide an external voltage for SPKVDD from the header pin.
- **AVDD-ISO** - By default, this jumper is closed. Cut this jumper to isolate AVDD from the voltage regulator to provide a VDD from the header pin.
- **LED** - By default, this jumper is closed and located on the bottom of the board. Cut this trace to disable the LED that is connected to the output of the 3.3V voltage regulator.
- **I2C** - By default, this 3-pad jumper is closed and located on the bottom of the board. The 2.2kΩ pull-up resistors are attached to the primary I^2C bus; if multiple devices are connected to the bus with the pull-up resistors enabled, the parallel equivalent resistance will create too strong of a pull-up for the bus to operate correctly. As a general rule of thumb, disable all but one pair of pull-up resistors if multiple devices are connected to the bus.

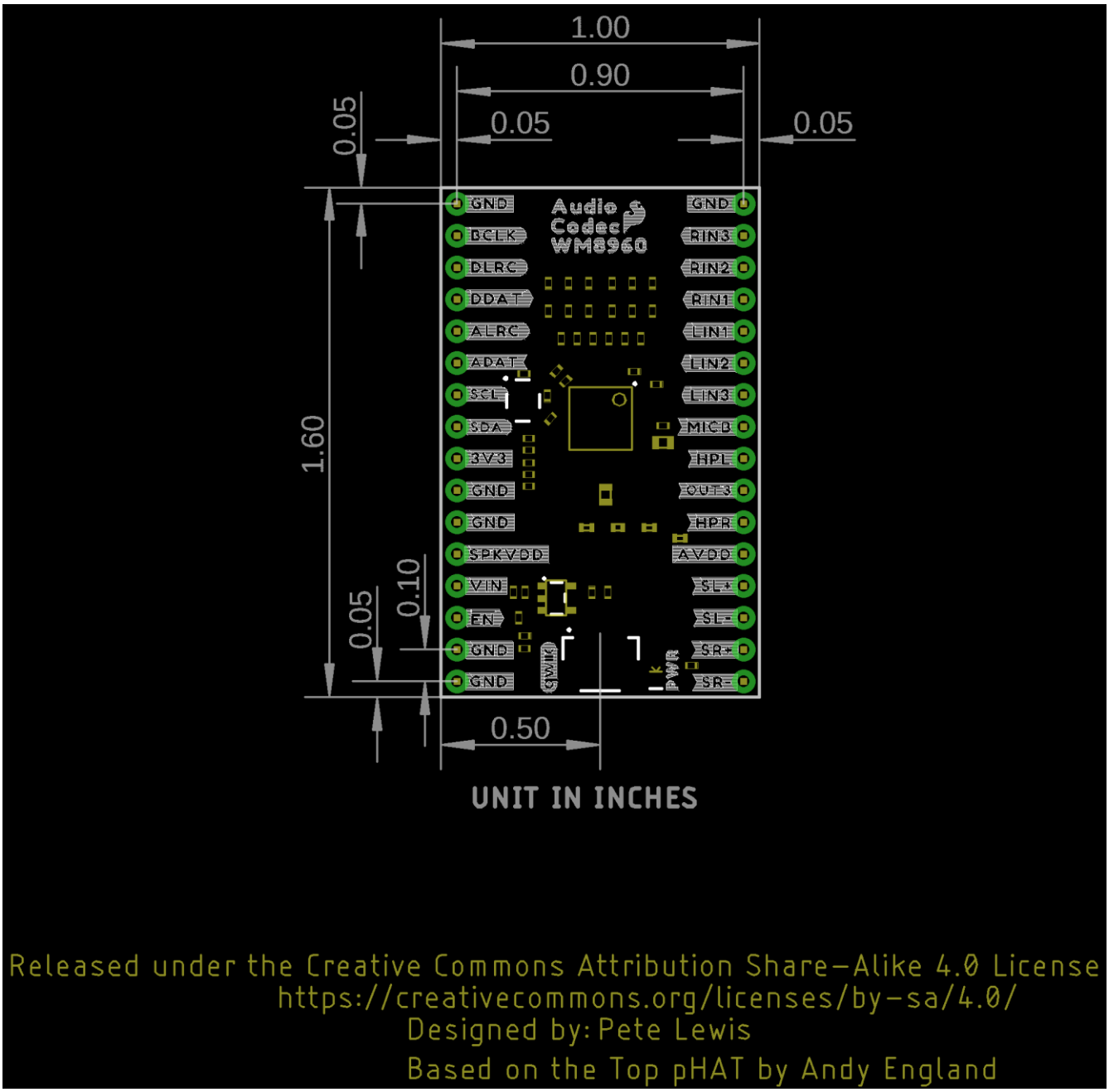


⚡ **Warning!** If you decide to cut the trace between **VDD** and **SPKVDD** to connect **SPKVDD** to **3V3**, make sure to avoid any shorts between **VDD** and **SPKVDD**. If you place an input voltage on **VIN** that is higher than **3V3** pin is rated for will damage the WM8960 and anything connected to **3V3**.

For users providing their own power supply for **SPKVDD** and **AVDD**, make sure to keep the jumpers open.

Board Dimensions

The board dimensions of the Audio Codec Breakout - WM8960 is 1.60" x 1.00".



Hardware Hookup

There are a number of ways to connect to the input and output pins of the WM8960. We will go over each .

Soldering

Header pins were left off the Audio Codec Breakout to allow users the flexibility of connecting any type of 0.1"-spaced header to the board. Depending on your connections, you may need to solder additional

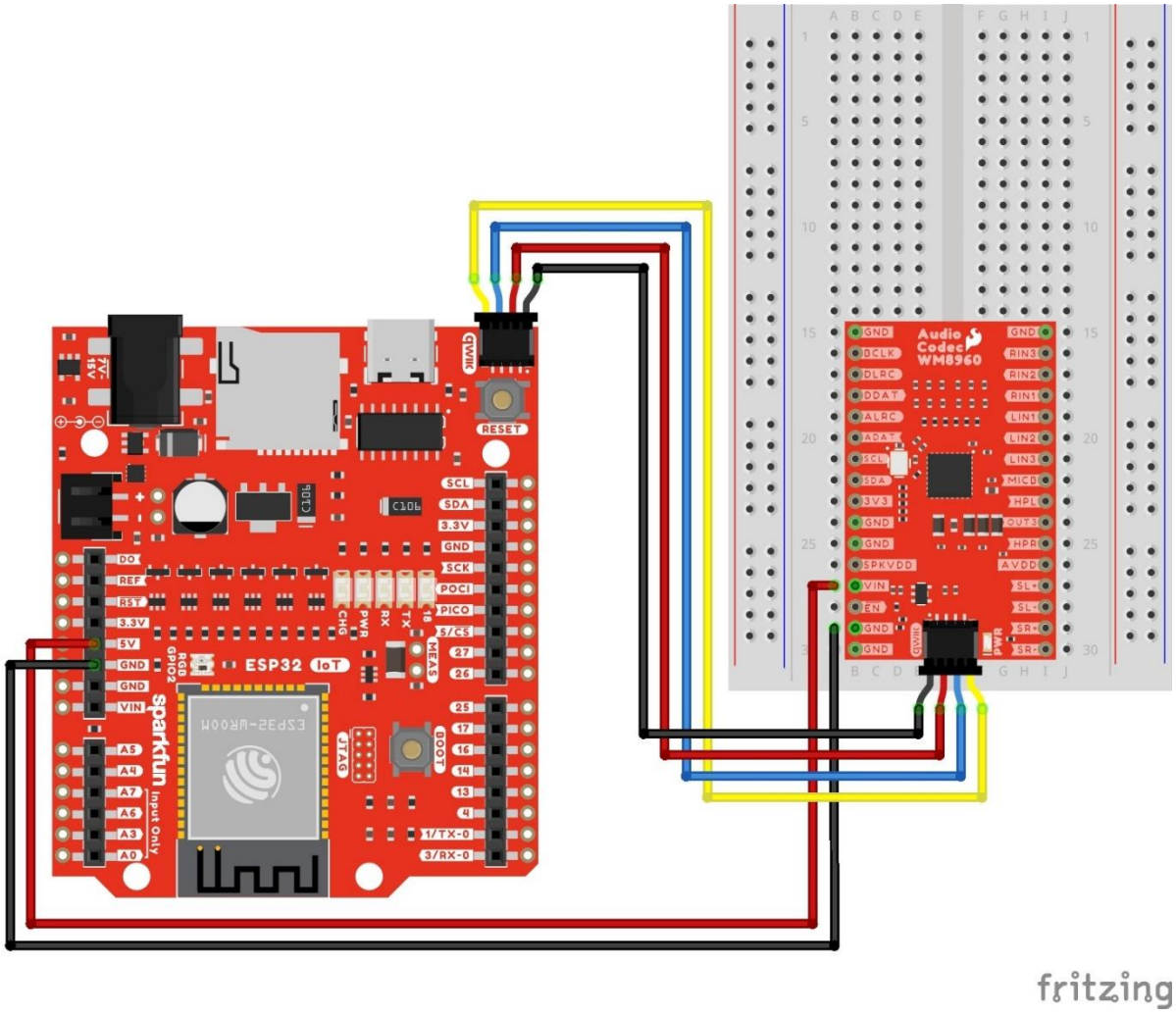
breakout boards and adapters. For temporary connections to the I/O pins, you could use IC hooks to test out the pins. However, you'll need to solder headers or wires of your choice to the board for a secure connection. For the scope of this tutorial, we will be soldering male header pins on the board and wiring the circuit on a breadboard. Here are a few tutorials to connect to the pads depending on your personal preference.

Power

To power the WM8960 appropriately, you will just need to provide power to **VIN** and the **3V3** pins. As stated in the Hardware Overview: Power, you can take advantage of your development board's voltages without adjusting jumpers. We recommend making the following connection listed in the table.

Audio Codec - WM8960	IoT RedBoard - ESP32
VIN	5V
GND	GND (Additional Ground, optional)
Qwiic Cable's 3.3V pin (or 3.3V)	Qwiic Cable's 3.3V pin (or 3.3V)
Qwiic Cable's GND pin (or GND)	Qwiic Cable's GND pin (or GND)

Using wires and the Qwiic cable to power VIN and 3V3, your circuit should look similar to the circuit diagram below. We will use this method to power the audio codec for the proceeding circuit diagrams.



⚡ **Danger:** Note that we are connecting 5V to the audio codec's VIN pin. Depending on the development board that you are using VIN from your Arduino may be too high of a voltage for the Audio Codec's VIN

pin. Exceeding the voltage beyond the absolue maximum rating can damage the components on the audio codec breakout.

Depending on your setup, you could provide a separate input voltage to the SPKVDD and VDD. Just make sure to adjust the jumpers on the back as necessary.

⚡ **Warning!** If you decide to cut the trace between **VDD** and **SPKVDD** to connect **SPKVDD** to **3V3**, make sure to avoid any shorts between **VDD** and **SPKVDD**. If you place an input voltage on **VIN** that is higher than **3V3** pin is rated for will damage the WM8960 and anything connected to **3V3**.

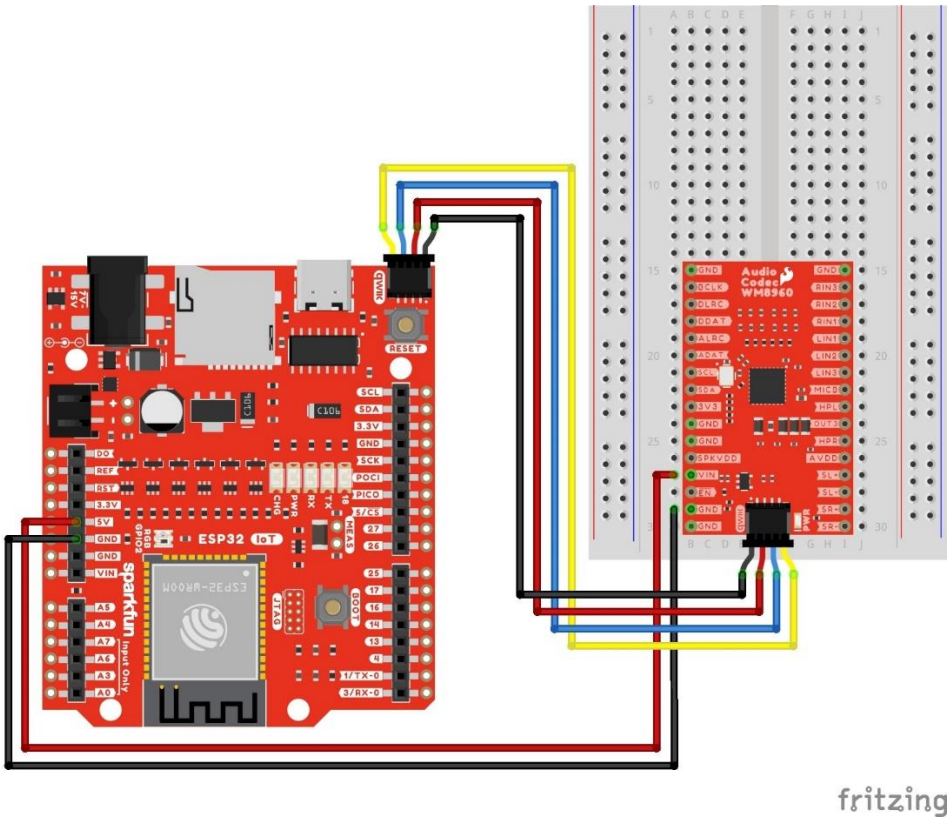
For users providing their own power supply for **SPKVDD** and **AVDD**, make sure to keep the jumpers open.

Qwiic and I²C

To configure the I²S settings or signal path, you will need to connect to the I²C port. Users can save some time wiring this part of the circuit up by adding a Qwiic cable between the audio codec breakout and the IoT RedBoard ESP32. As an alternative, users could also solder to the PTH as well. We recommend making the following connection listed in the table.

Audio Codec - WM8960	IoT RedBoard - ESP32
Qwiic Cable's SCL pin (or SCL)	Qwiic Cable's SCL pin (or SCL)
Qwiic Cable's SDA pin (or SDA)	Qwiic Cable's SDA pin (or SDA)
Qwiic Cable's 3.3V pin (or 3.3V)	Qwiic Cable's 3.3V pin (or 3.3V)
Qwiic Cable's GND pin (or GND)	Qwiic Cable's GND pin (or GND)

Surprise! Your connection is basically the same as the circuit diagram in the previous section. We just highlighted the circuit connection for I²C data and clock lines here.

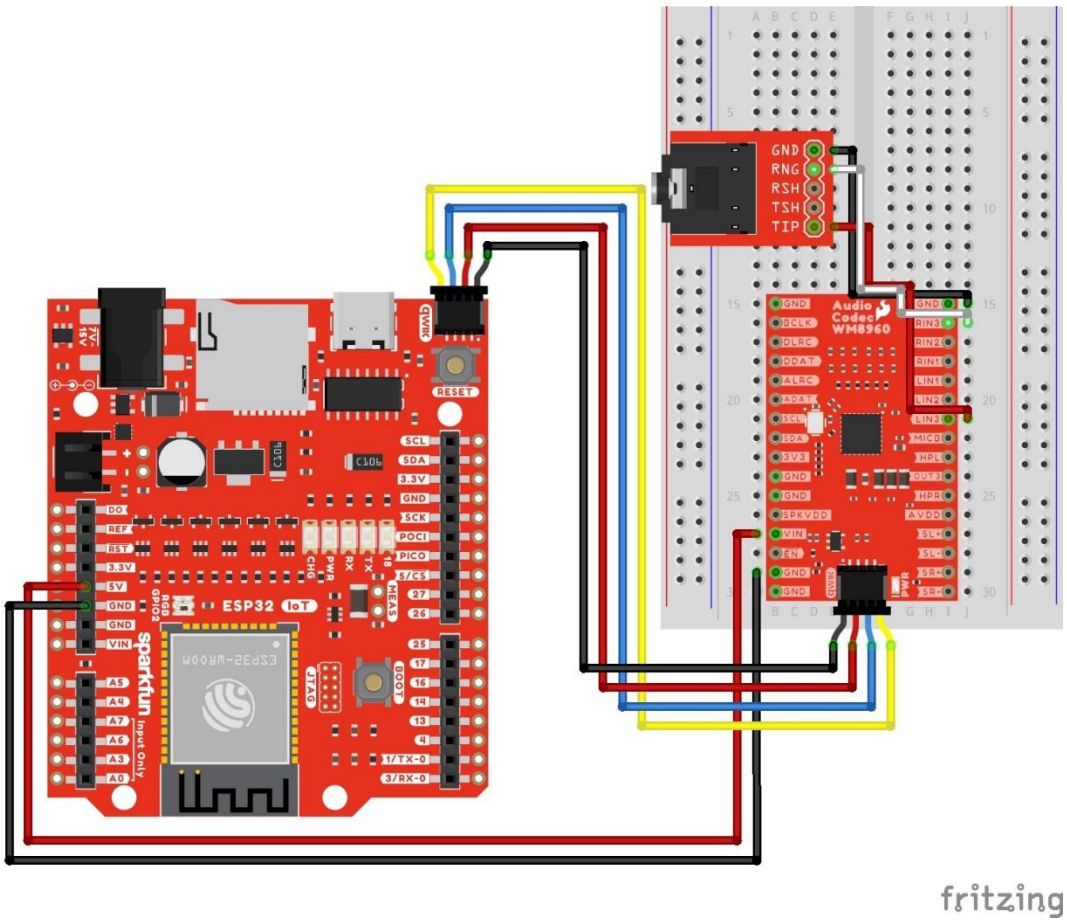


Line Input 3

The following connection is for a line level input 3. For simplicity, the table does not include the power and I^2C pins.

Audio Codec - WM8960	TRS Connector
LINPUT3	Tip
RINPUT3	Ring
GND	Sleeve

Your circuit should look similar to the circuit diagram below.

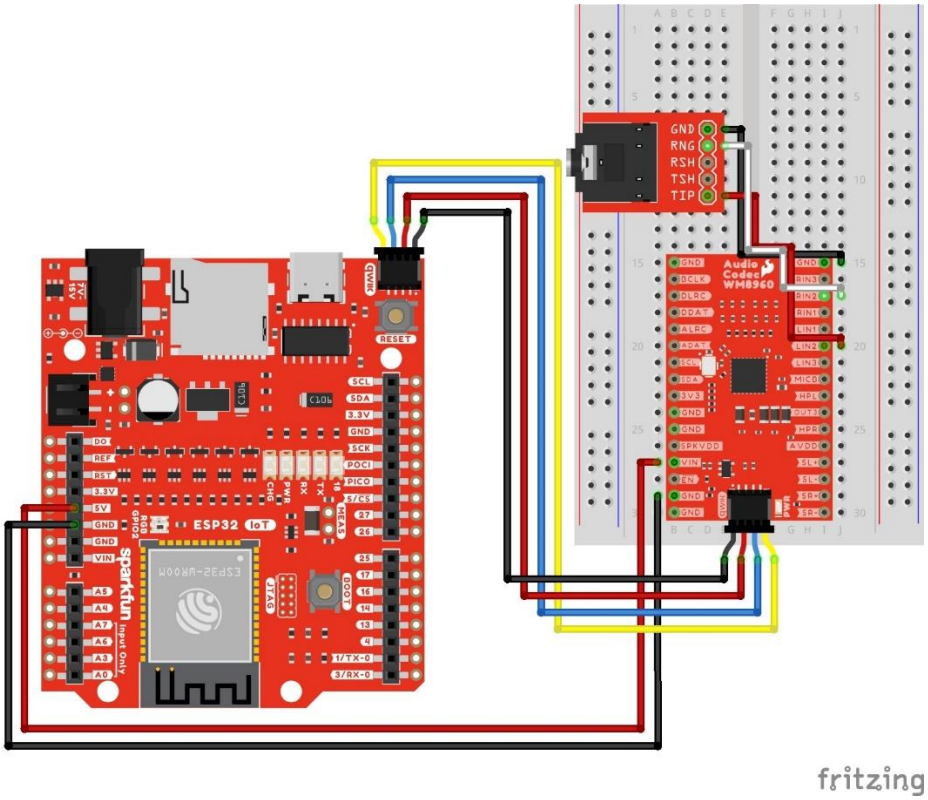


Line Input 2

The following connection is for a line level input 2. For simplicity, the table does not include the power and I^2C pins.

Audio Codec - WM8960	TRS Connector
LINPUT2	Tip
RINPUT2	Ring
GND	Sleeve

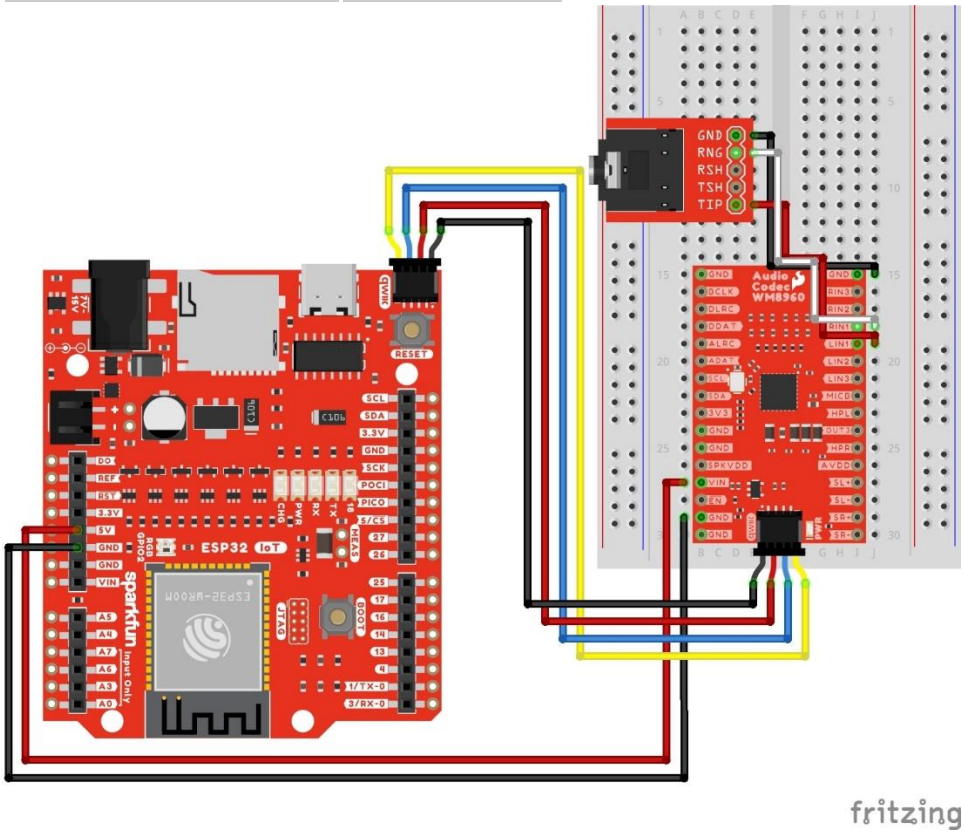
Your circuit should look similar to the circuit diagram below.



Line Input 1

The following connection is for a line level input 1. For simplicity, the table does not include the power and I^2C pins. Your circuit should look similar to the circuit diagram below.

Audio Codec - WM8960	TRS Connector
LINPUT1	Tip
RINPUT1	Ring
GND	Sleeve

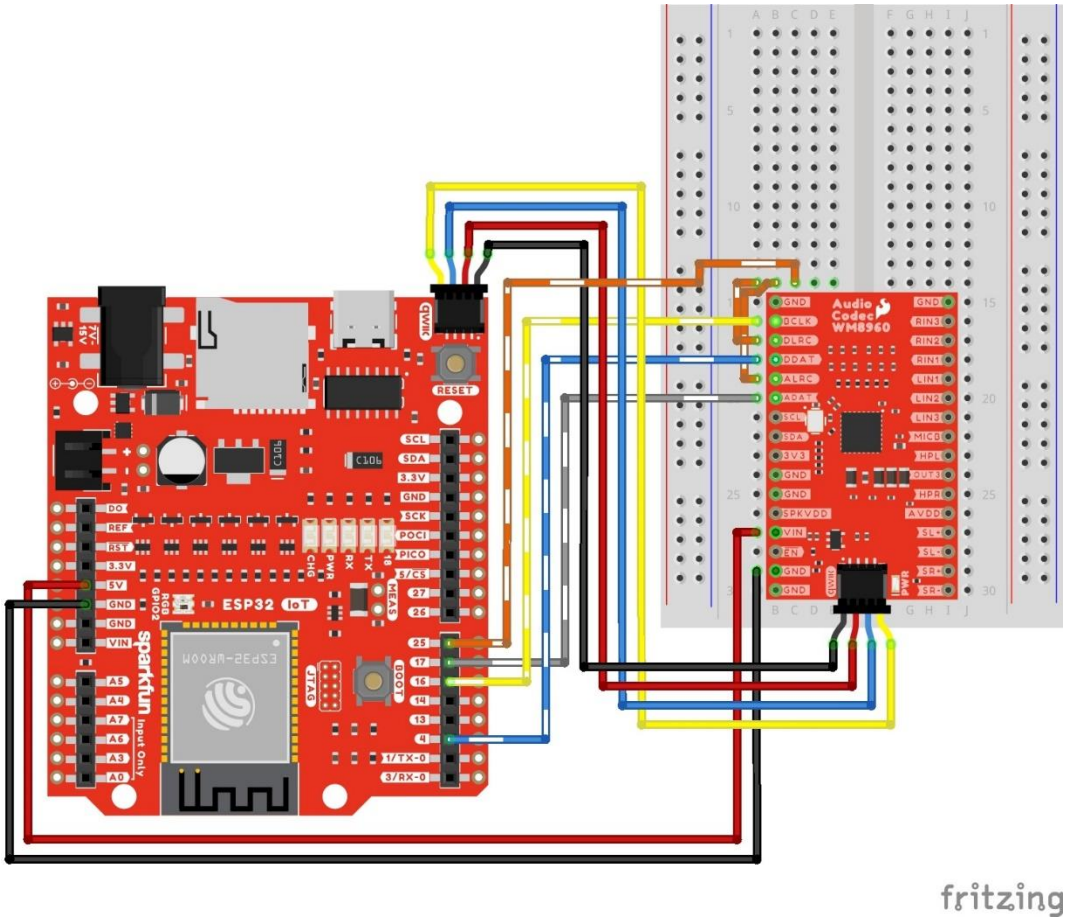


I²S Passthrough

The following connection is for passing an audio source through the ADC and immediately back to the DAC. For simplicity, the table does not include the power and I²C pins. Note that this is for setting the codec as a I²S peripheral.

Audio Codec - WM8960	IoT RedBoard - ESP32
DACDAT	4
BCLK	16
ADCDAT	17
DACLRC	25
ADCLRC	25

Your circuit should look similar to the circuit diagram below.

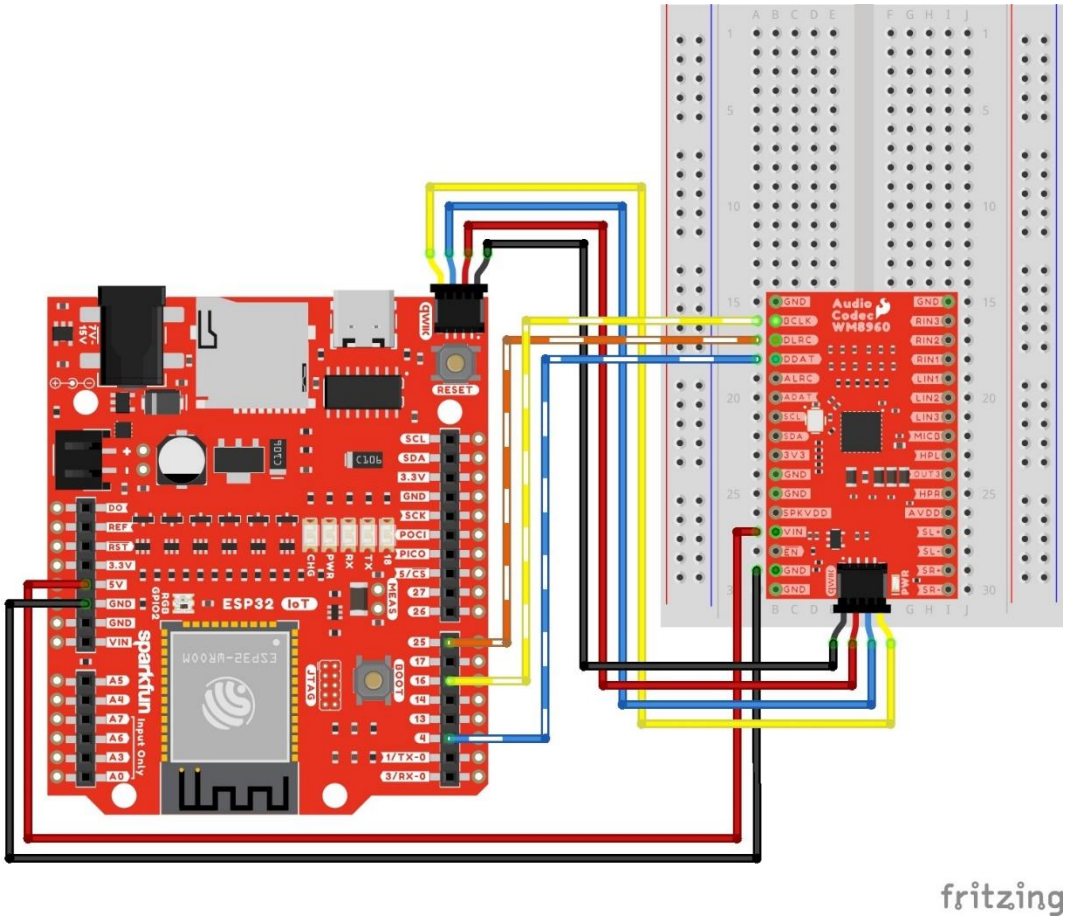


I²S Decoder

The following connection is for decoding audio. One example is if users connect an audio Bluetooth® device (such as your phone or laptop) to the ESP32 and stream music wirelessly. For simplicity, the table does not include the power and I²C pins. Note that this is for setting the codec as a I²S peripheral.

Audio Codec - WM8960	IoT RedBoard - ESP32
DACDAT	4
BCLK	16
DACLRC	25

Your circuit should look similar to the circuit diagram below.

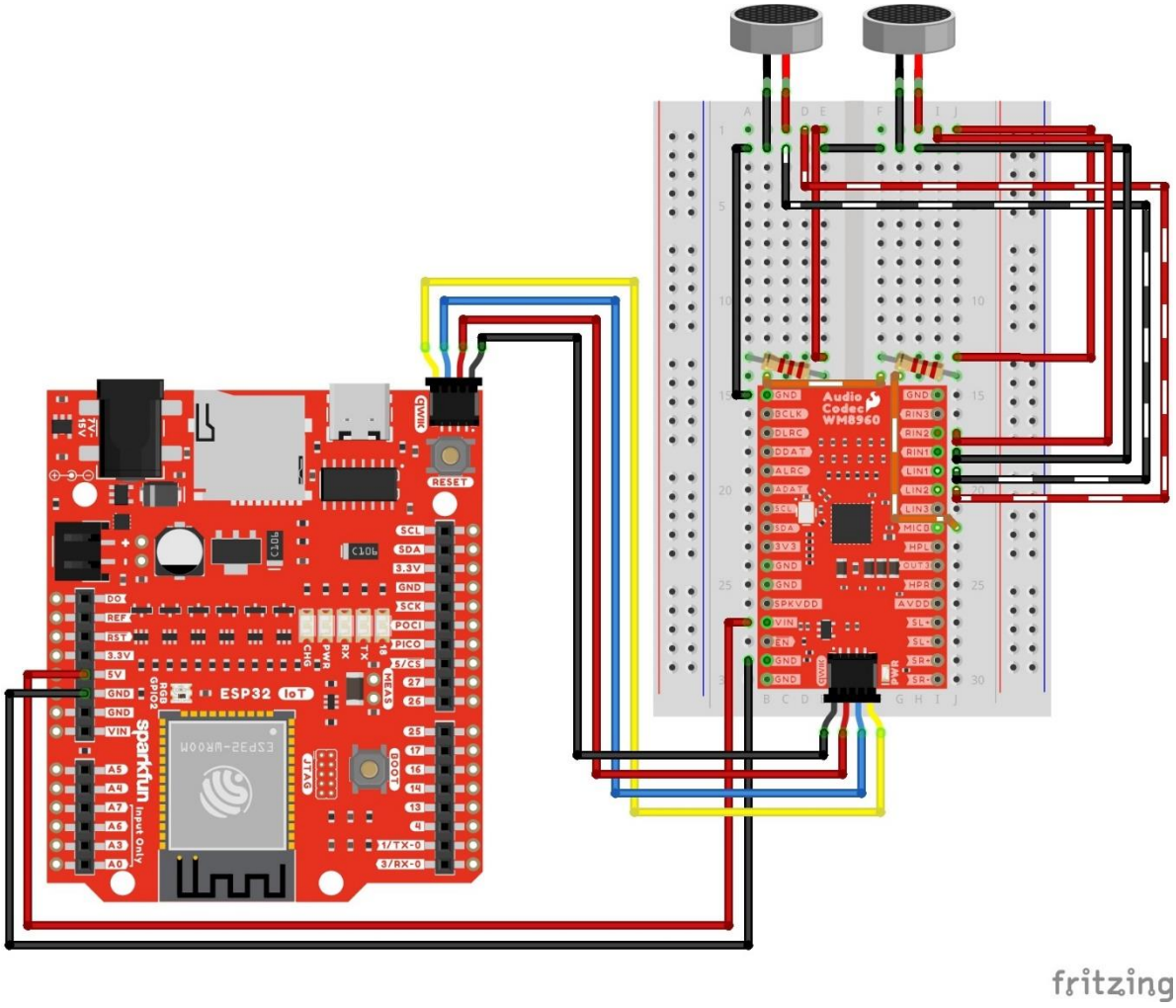


Differential Microphone Input

The following connection is for connecting differential microphones to the audio codec. When using a pseudo-differential microphone configuration, make sure to include a 2.2kΩ resistor between the MICBIAS and the + terminals of each differential microphone. You will also need to set voltage on the MICBIAS pin. For simplicity, the table does not include the power and I^2C pins.

Audio Codec - WM8960	Resistor	Electret Microphone
GND		Left Mic -
LINPUT1		Left Mic -
LINPUT2		Left Mic +
MICBIAS	2.2kΩ	Right Mic +
GND		Right Mic -
LINPUT1		Right Mic -
LINPUT2		Right Mic +
MICBIAS	2.2kΩ	Right Mic +

Your circuit should look similar to the circuit diagram below.

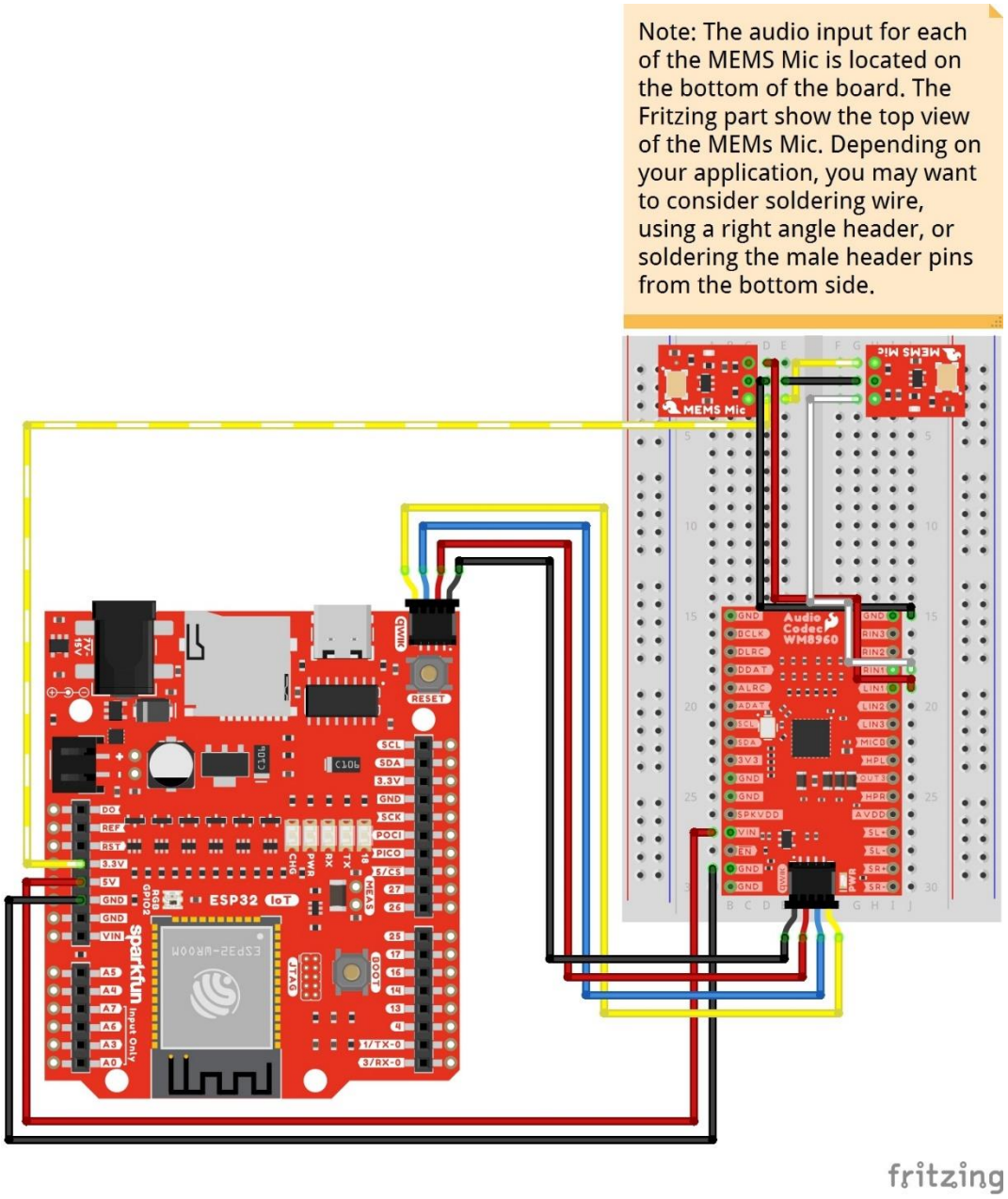


Single Ended Microphone Input

The following connection is for connecting single ended microphones to the audio codec. Instead of using a line level input from the TRS connector, users can also connect unbalanced microphones (these are usually microphones with an amplifier on the breakout boards and have one audio output pin) to the left and right pins of INPUT1. Make sure to also connect power and GND to the boards. For simplicity, the table does not include the power and I^2C pins between the ESP32 and WM8960.

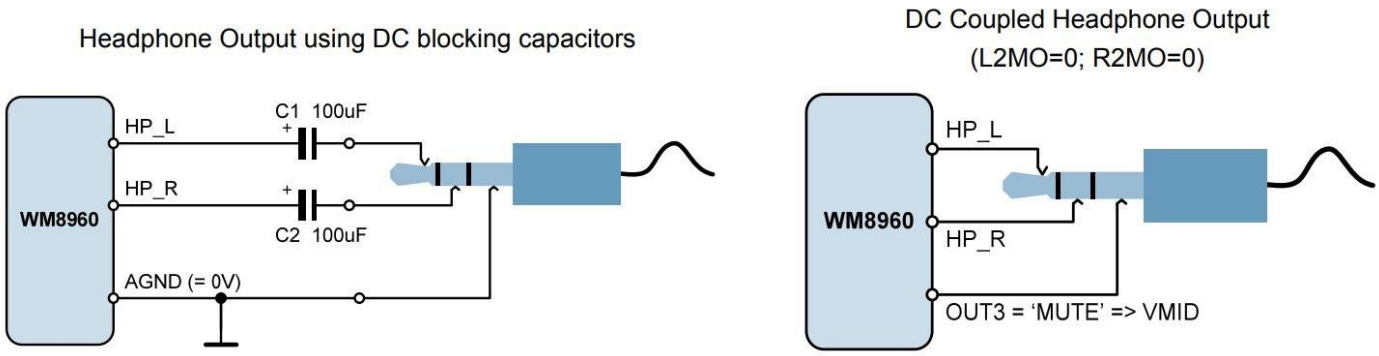
Audio Codec - WM8960	Single Ended Microphone (Left)	Single Ended Microphone (Right)	IoT RedBoard - ESP32
Qwiic Cable's GND pin (or GND)	GND	GND	GND
LINPUT1	AUD		
RINPUT1		AUD	
Qwiic Cable's 3.3V pin (or 3.3V)	3.3V	3.3V	3.3V

Your circuit should look similar to the circuit diagram below.



Headphone Output

Based on the recommended output configuration from the datasheet, we will be using a capless headphone output in this tutorial.

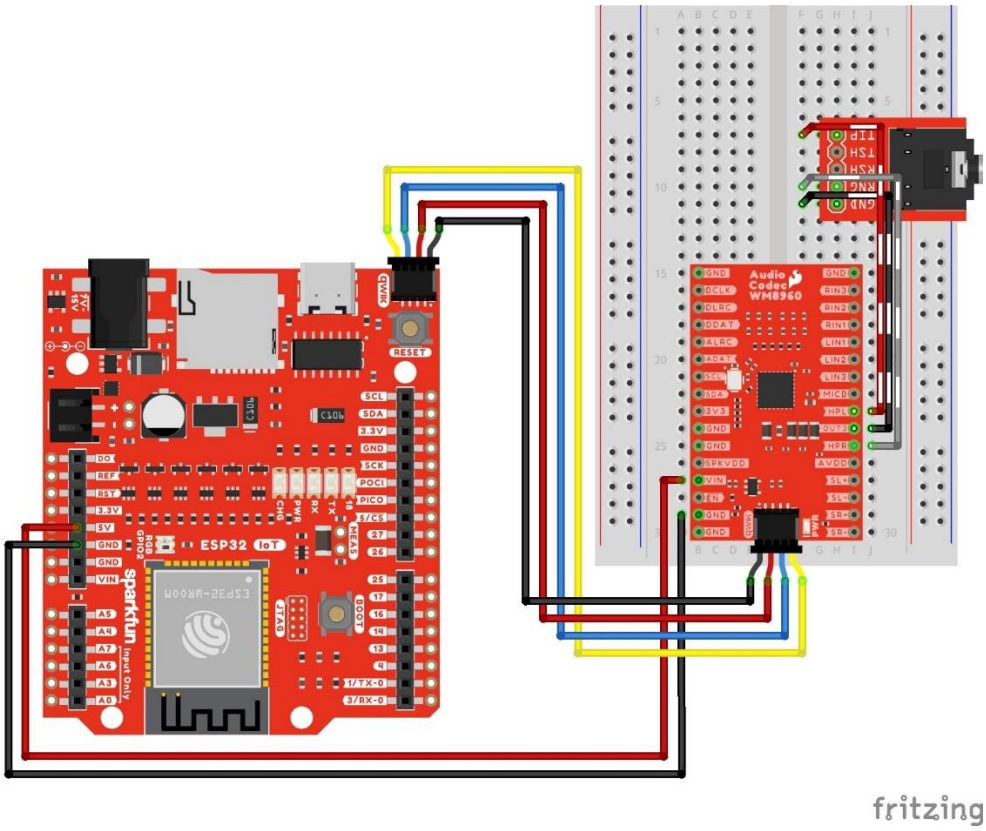


Recommended headphone output configuration from the WM8960 Datasheet (v4.2) on page 41.

The following connection is for connecting headphones to the audio codec's headphone output. For simplicity, the table does not include the power and I^2C pins.

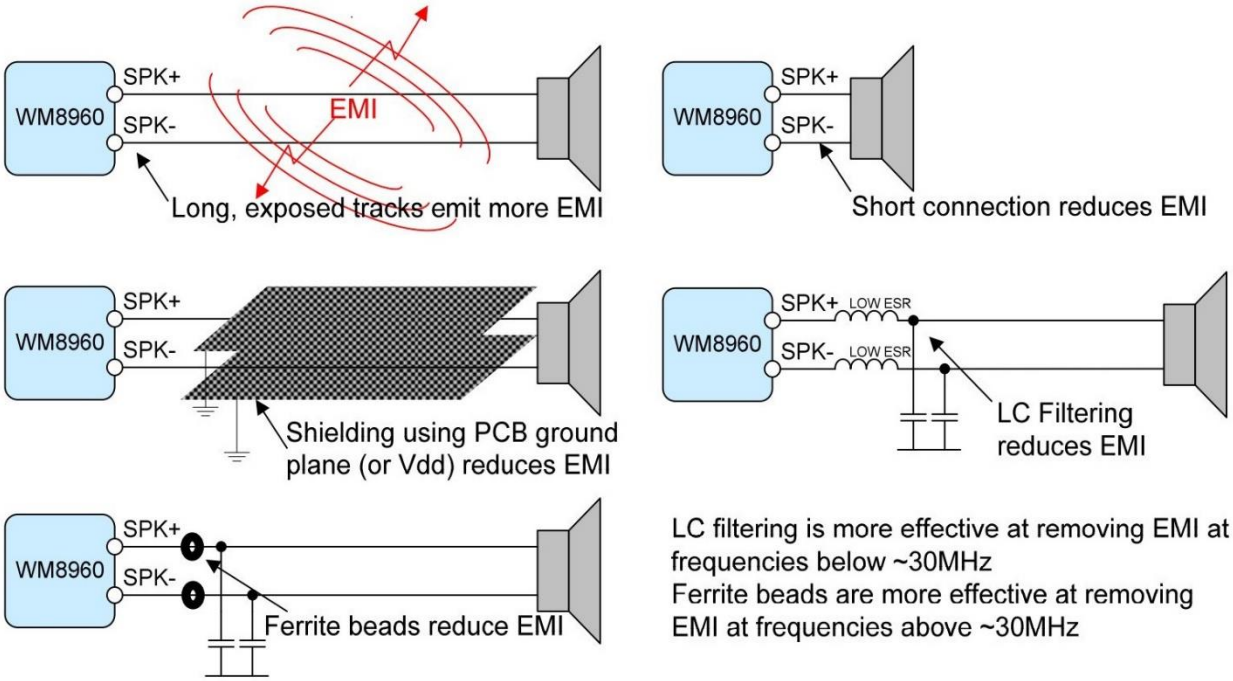
Audio Codec - WM8960 TRS Connector

HPL	Tip
HPR	Ring
OUT3	Sleeve



Differential Speaker Output

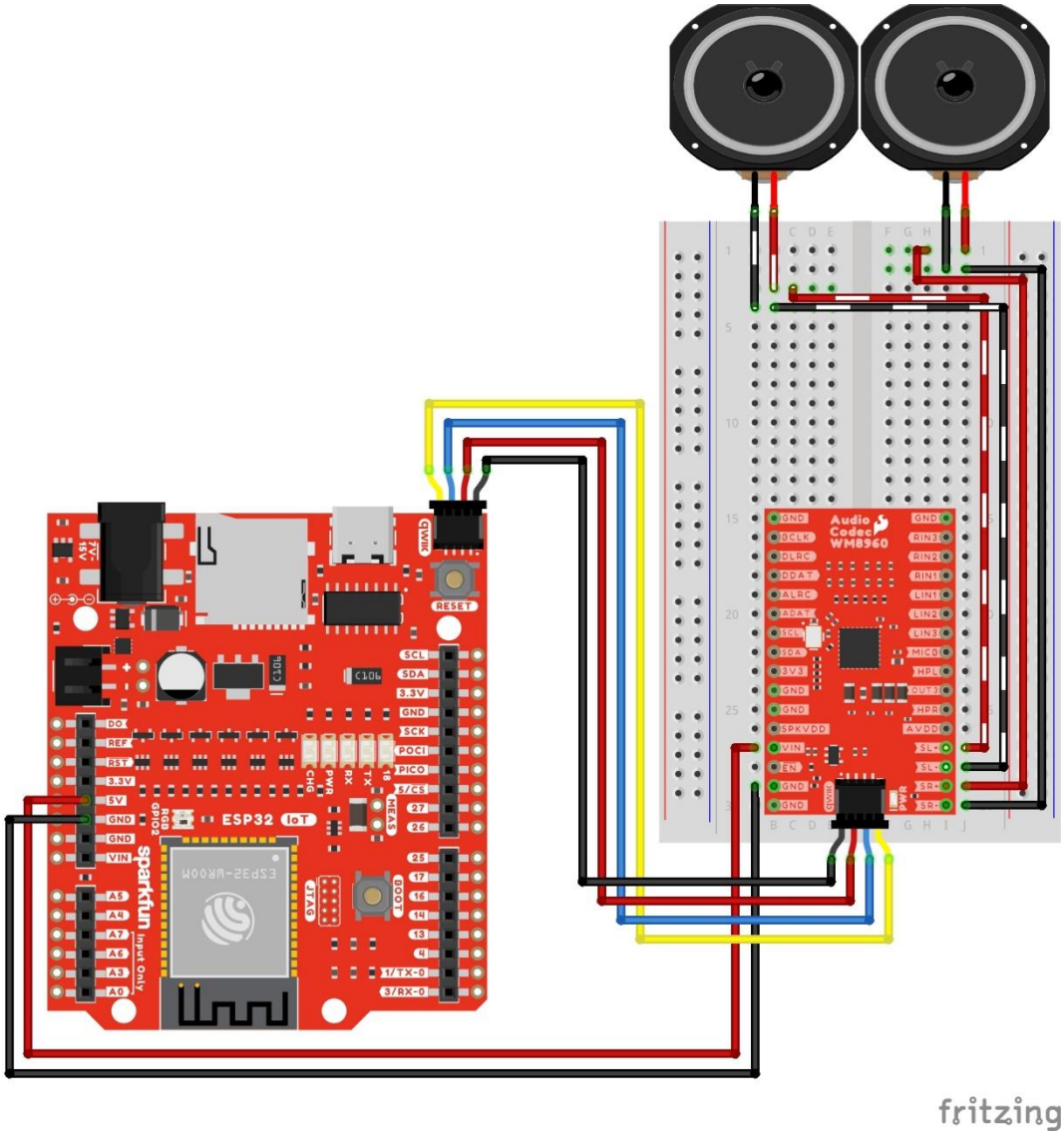
There are a few recommended speaker output configurations from the datasheet to minimize the speaker connection losses due to series resistance and EMI. For a basic setup, we will be using a short wire connection between the audio codec's breakout board and each differential speaker. For those that require long exposed track and want to go the extra mile, users can build an LC filter circuit, add ferrite beads, or shield the wires using PCB ground plane (or Vdd).



Recommended speaker output configuration from the WM8960 Datasheet (v4.2) on page 88.

The following connection is for connecting speakers to the audio codec's speaker output channels. For simplicity, the table does not include the power and I^2C pins.

Audio Codec - WM8960	Speakers
SL+	Speaker Left +
SL-	Speaker Left -
SR+	Speaker Right +
SR+	Speaker Right -



Software Installation

Note: This example assumes you are using the latest version of the Arduino IDE on your desktop. If this is your first time using Arduino, please review the following tutorials.

- Installing the Arduino IDE
- Installing Board Definitions in the Arduino IDE
- Installing an Arduino Library

Arduino Board Definitions and Driver

We'll assume that you installed the necessary board files and drivers for your development board. In this case, we used the IoT RedBoard - ESP32 which uses the CH340 USB-to-serial converter. If you are using a Processor Board, make sure to check out its hookup guide for your Processor Board.

How do I install a custom Arduino board/core? It's easy! This tutorial will go over how to install an Arduino board definition using the Arduino Board Manager. We will also go over manually installing third-party cores, such as the board definitions required for many of the SparkFun development boards.

Installing the Arduino Library

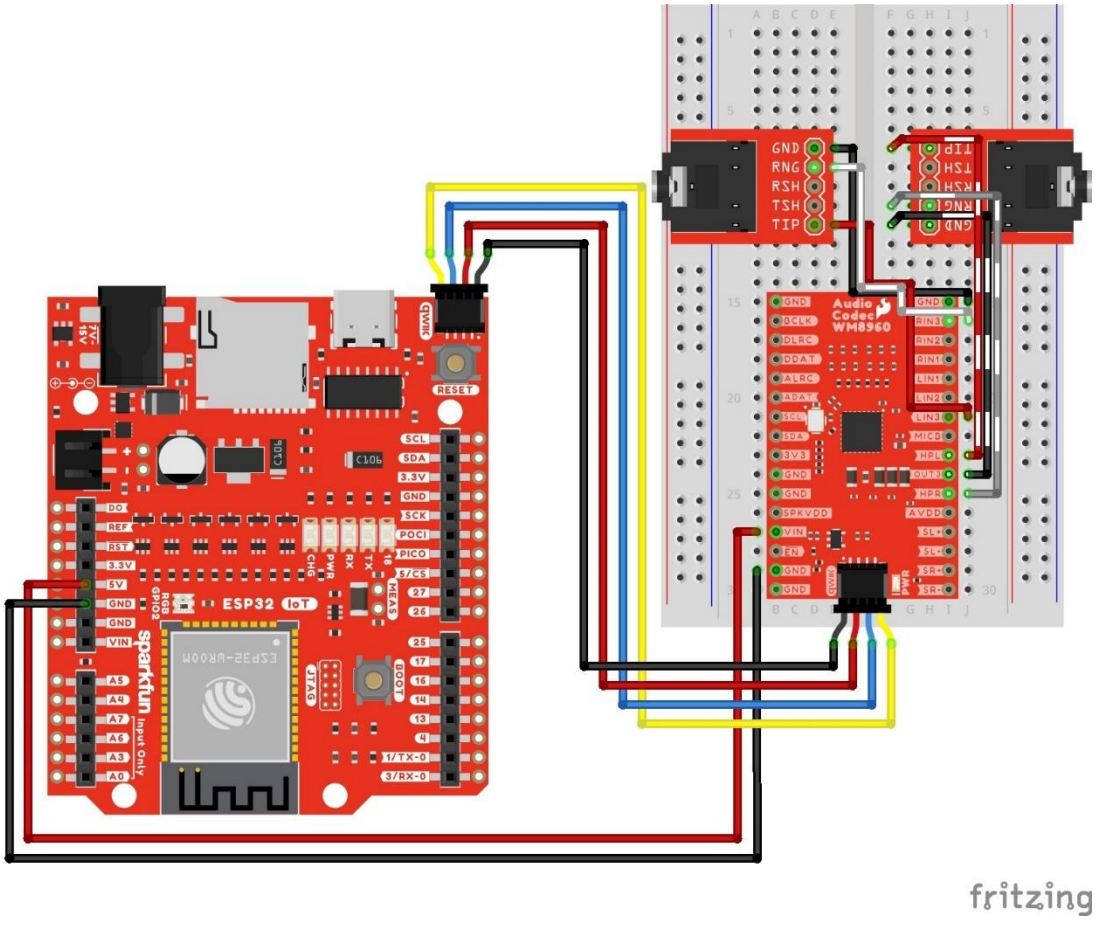
The SparkFun Arduino library can be downloaded with the Arduino library manager by searching '**SparkFun Audio Codec Breakout WM8960**' or you can grab the zip here from the GitHub repository to manually install.

Example 1: Volume

In this example, we will pass a line level audio source into the WM8960's line input 3 port. The signal will go through the mixers and gain stages of the audio codec. For this output, we will be sending the audio to the headphones.

Hardware Hookup

Connect power, I^{2}C, line input 3 port, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



Connect a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_01_Volume**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones. After every 5 seconds, the volume will decrease until it is muted.

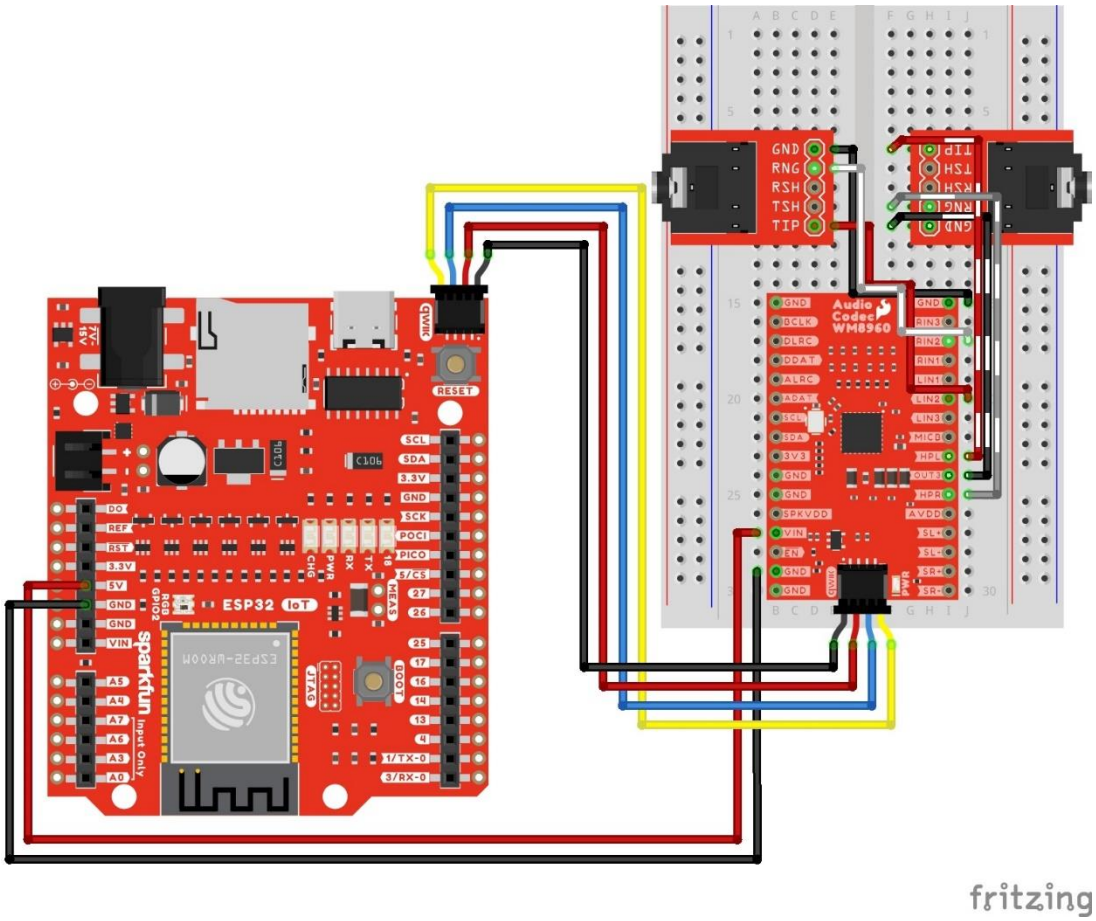
If you waited too long to hit the play button, the audio codec may have already muted the track. Try hitting the reset button on the IoT RedBoard to restart the example since it executes the code once.

Example 2: Line Input 2

In this example, we will pass a line level audio source into the WM8960's line input 2 port. The signal will go through the mixers and gain stages of the audio codec like the previous example. For the output we will be sending the audio to the headphones as well.

Hardware Hookup

Connect power, I^{2}C, line input 2 port, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_02_INPUT2**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

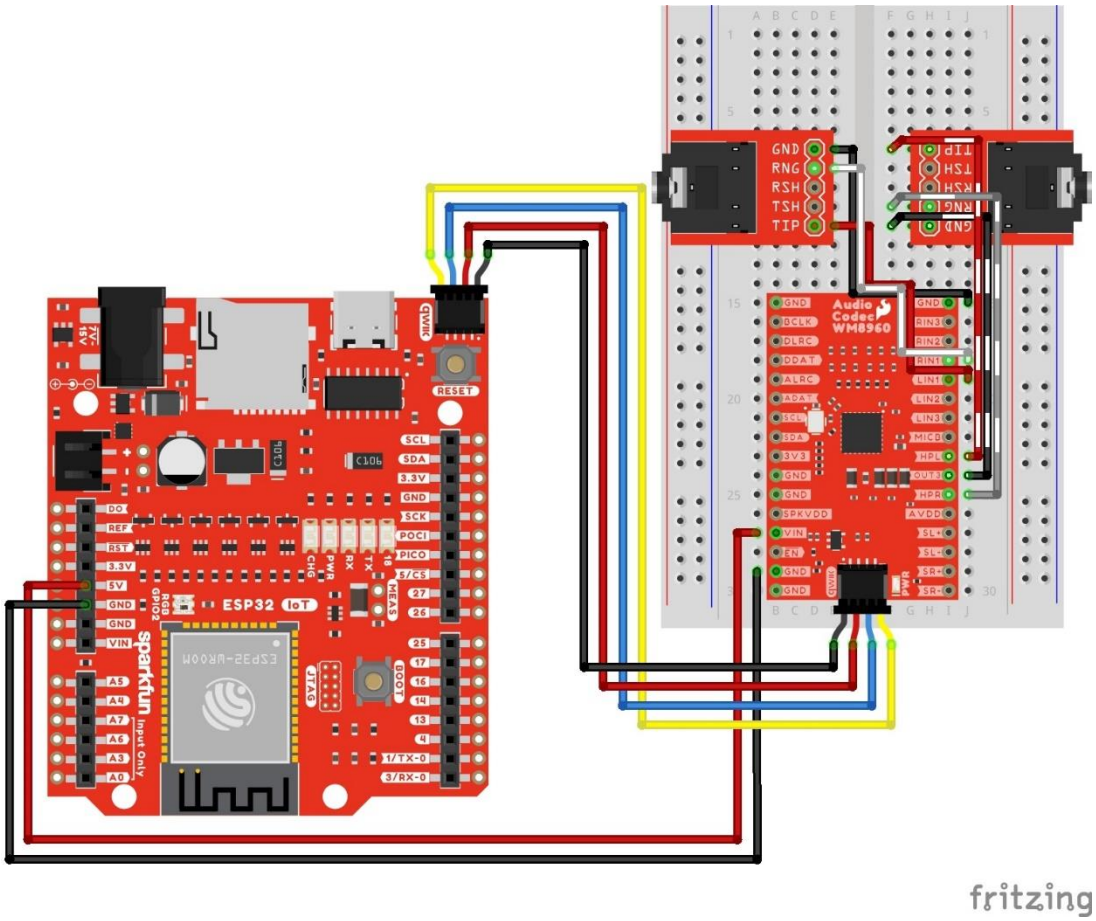
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones. Compared to the first example, this just sets the volume once.

Example 3: Line Input 1 [or Single Ended Microphone Input]

Similar to the past two examples, we will pass the line level audio source into the WM8960's line input 1 port. As an alternative, users can also wire up single ended microphones to input 1 instead of using a line level audio source from the TRS connector. The signal will go through the mixers and gain stages of the audio codec. Again, we will be sending the audio to the headphone output.

Hardware Hookup 1

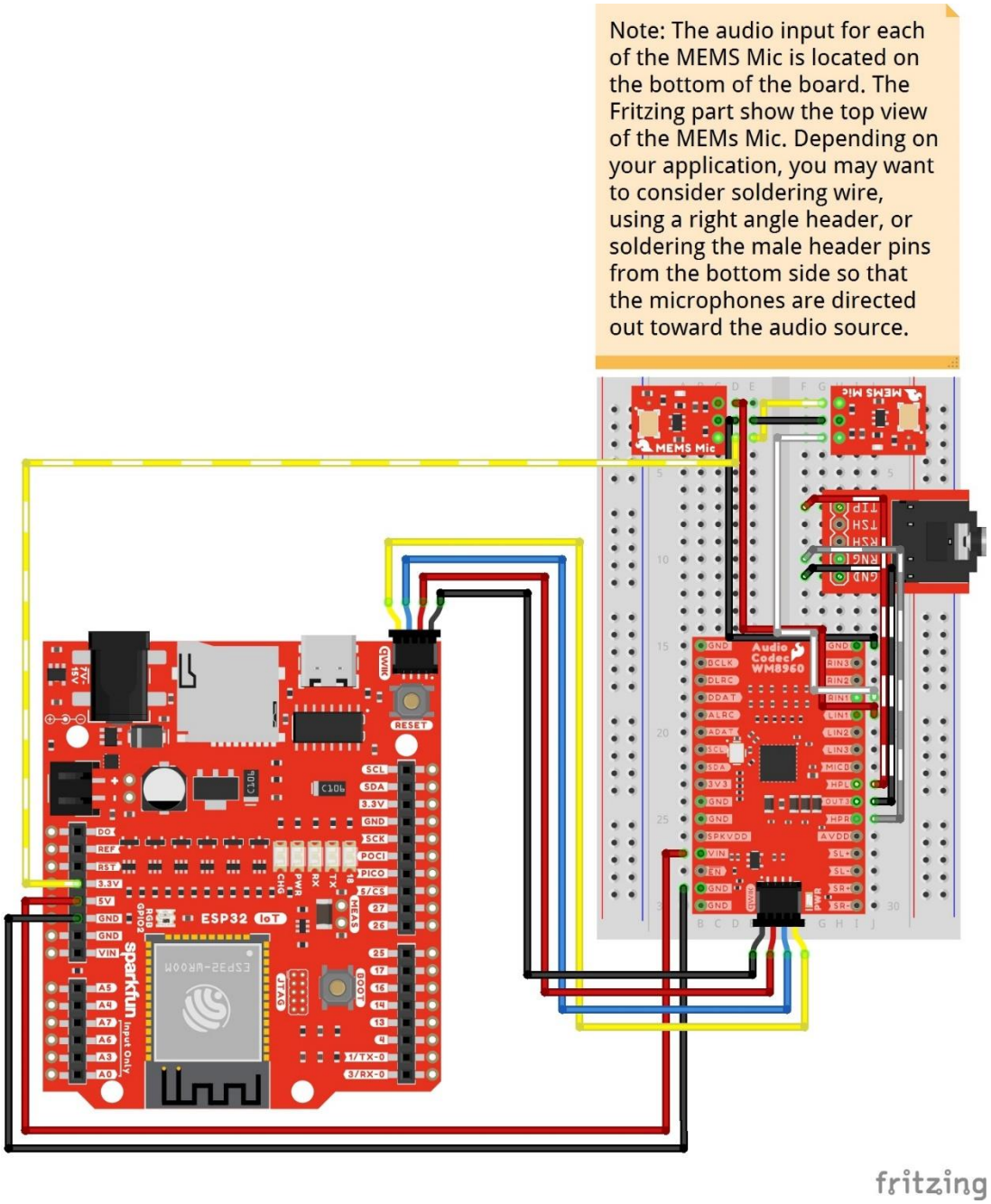
Connect power, I^{2}C, line input 1, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



Connect a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Hardware Hookup 2

Input 1 also has the ability to allow single ended microphones to the left and right inputs. As an alternative, you can also connect single ended electret or MEMS microphones to these pins. Connect power, I^2C, single ended microphones, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



Note: The audio input for each of the MEMS Mic is located on the bottom of the board. The Fritzing part show the top view of the MEMs Mic. Depending on your application, you may want to consider soldering wire, using a right angle header, or soldering the male header pins from the bottom side.

Connect a USB cable into your IoT RedBoard ESP32. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_03_INPUT1**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

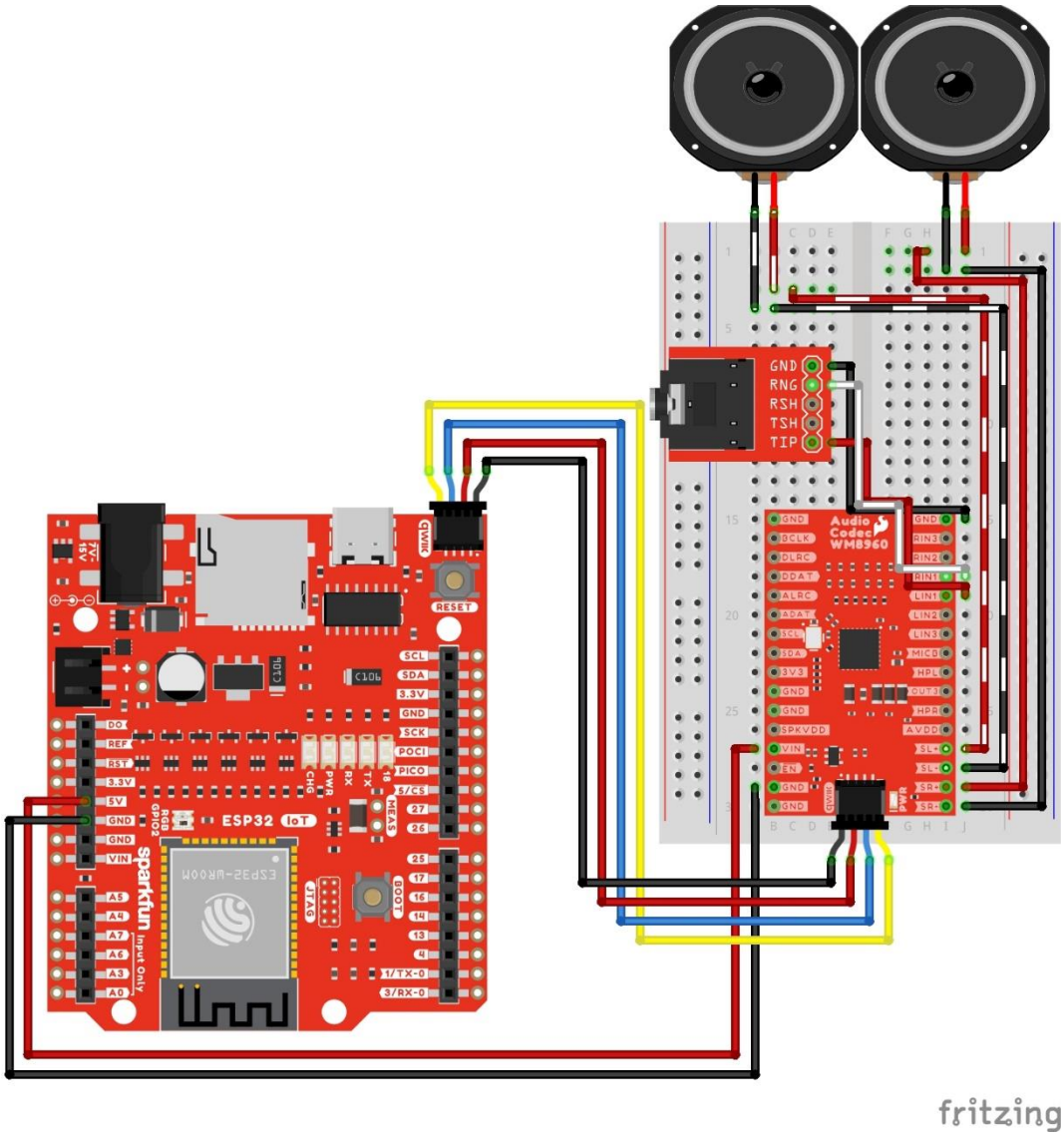
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source if you are using a TRS connector. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones. This is basically the same as the previous two examples but now we are using input 1.

Example 4: Speaker

In this example, we will pass a line level audio source into the WM8960's line input 1 port. The signal will go through the mixers and gain stages of the audio codec. For the output, we will be sending the audio to a pair of differential speakers.

Hardware Hookup

Connect power, I^{2}C, line input 1, and the speakers on the output channels as explained earlier. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_04_Speaker**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

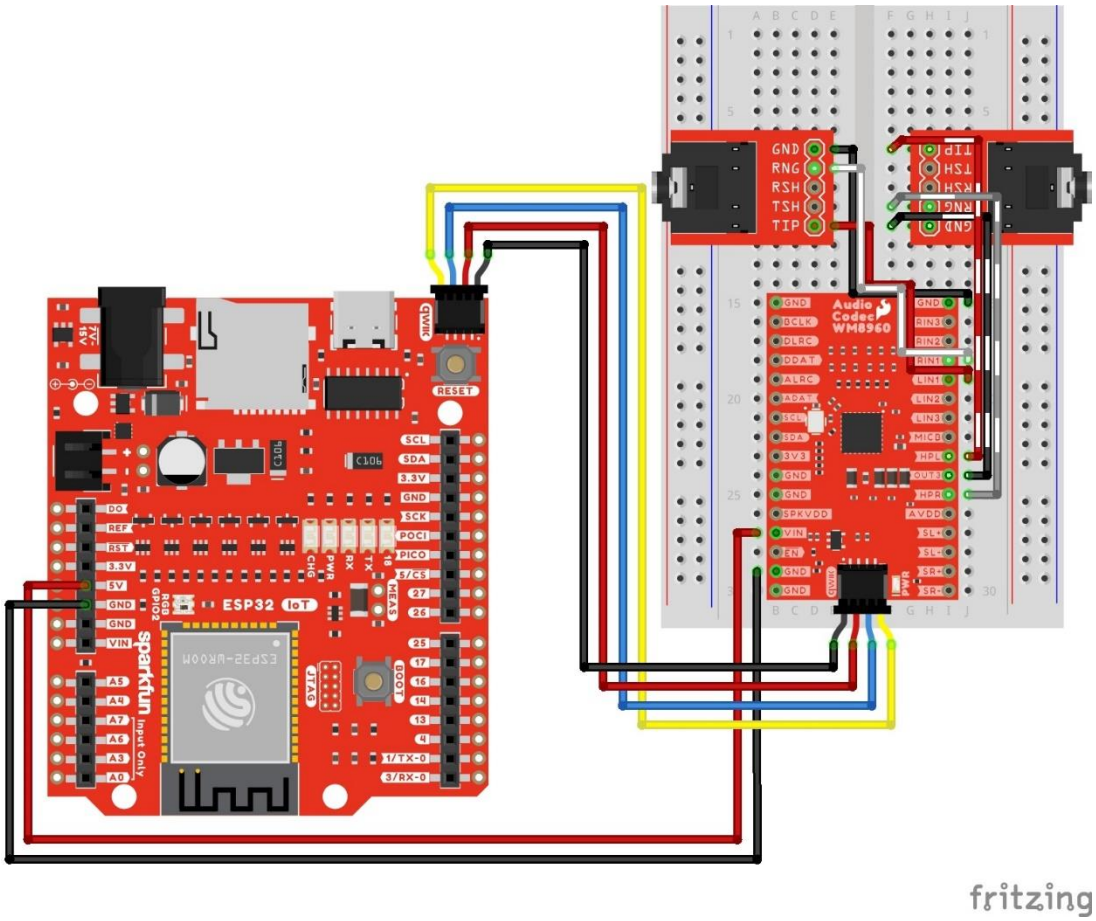
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the speakers.

Example 5: Loopback

In this example, we will pass a line level audio source into the WM8960's line input 1 port. The signal will go through the mixers and gain stages of the audio codec. What's different in this example is that we will also turn on loopback so that the ADC is fed directly to the DAC. The output of the DAC will then be sent to the headphone output.

Hardware Hookup

Connect power, I^2C, line input 1, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_05_Loopback**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

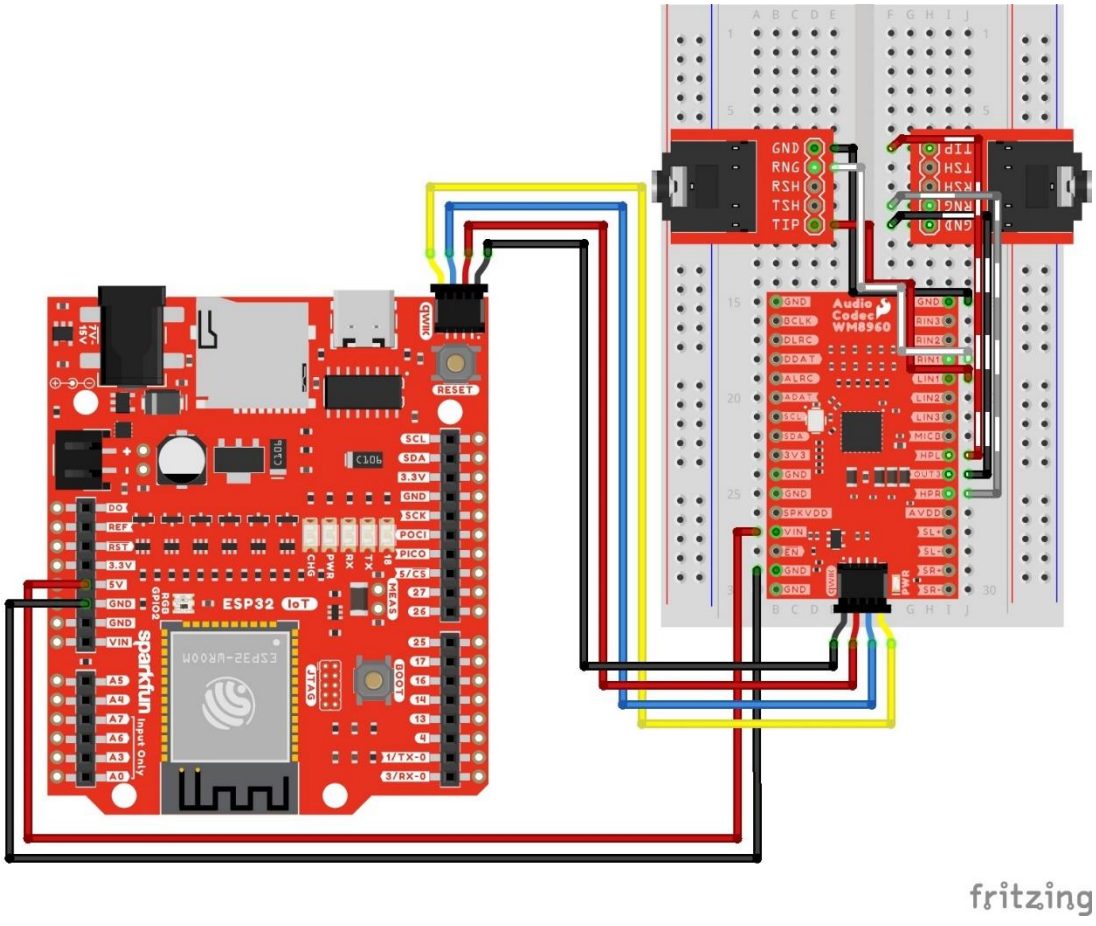
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones.

Example 6: 3D Enhance

In this example, we will pass a line level audio source into the WM8960's line input 1 port. The audio source will go through the mixers and gain stages into the ADC. With the loopback turned on, the ADC is sent directly to the DAC and output to the headphone output. In the `loop()` function, turn on and off the 3D enhance feature every 5 seconds.

Hardware Hookup

Connect power, I^{2}C, line input 1, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



Connect a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_06_3D_Enhance**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

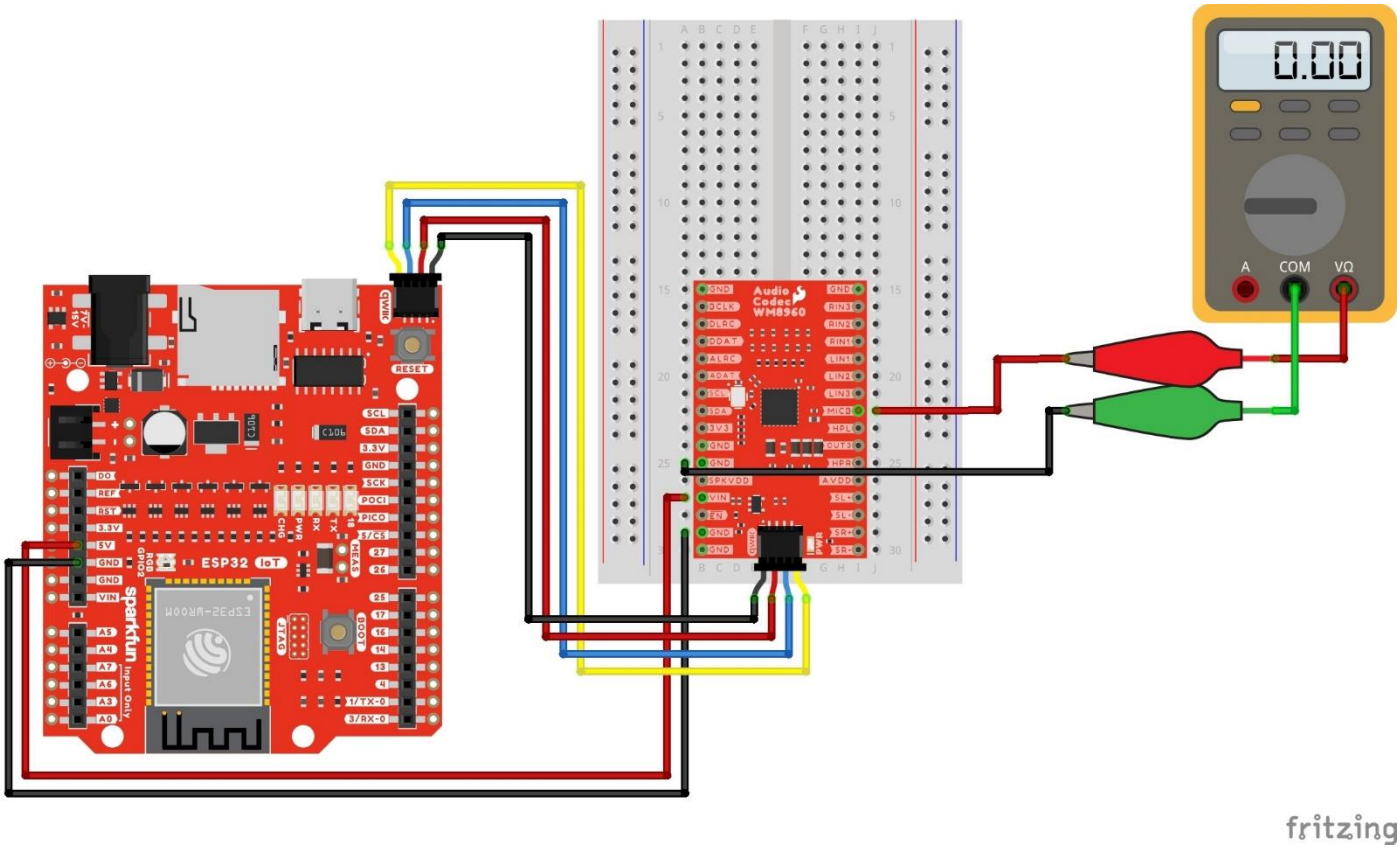
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output when the 3D enhance mode is enabled. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. Experience the virtual surround sound!

Example 7: Microphone Bias

in this example, we will set the microphone bias voltage and measure the voltage. This is for advanced users looking to add a differential microphone to the microphone input pins.

Hardware Hookup

Connect power and I^{2}C. as explained earlier. Then grab a multimeter with alligator clips and M/M jumper wires. Connect the wires to MICBIAS and GND to measure the voltage. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_07_MicBias**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Measuring the output with a multimeter, the voltage should be similar to the voltages that were set for the MICBIAS pin. After 3 seconds, the pin will be disabled. If you missed the window to measure the MICBIAS voltage, hit the reset button the IoT RedBoard to run the example again.

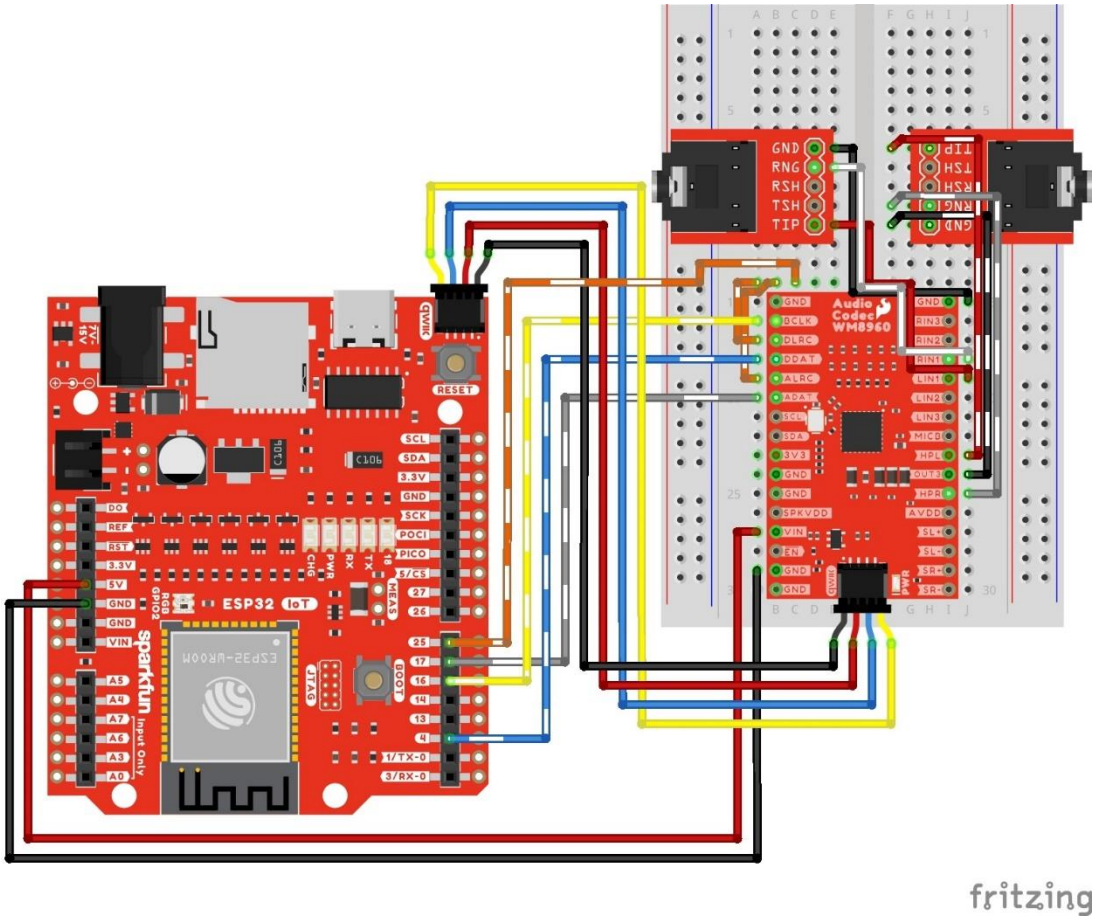
If you are satisfied with your MICBIAS voltage, head over to example 14 to add a resistor in series with each electret microphone.

Example 8: I2S Passthrough

In this example, we will pass a line level audio source into the WM8960's line input 1 port. The signal will go through the mixers and gain stages of the codec. The audio will be read from the ADC via I^{2}S and then immediately sent back to the DAC via I^{2}S. The output of the DAC will then be sent to the headphone output.

Hardware Hookup

Connect power, I^{2}C, line input 3, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



Connect a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_08_I2S_Passthrough**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones.

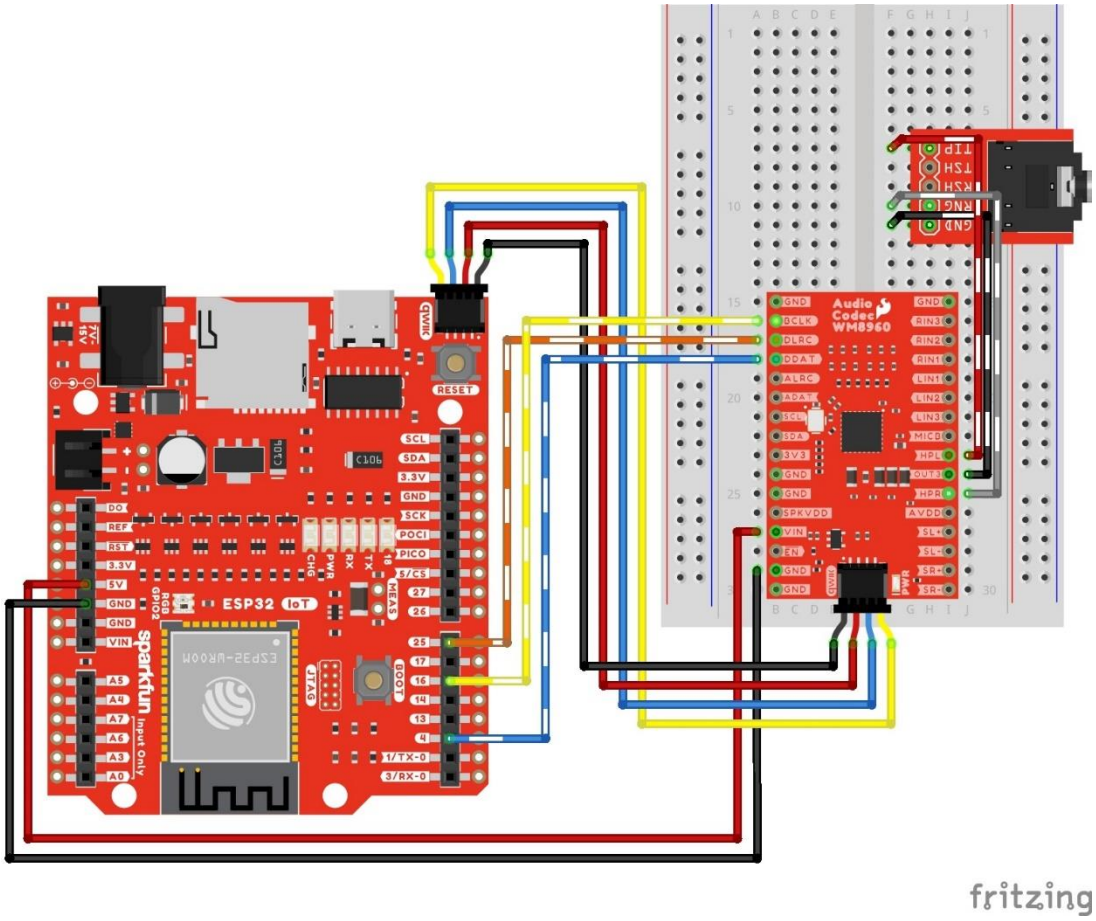
Example 9: I2S Bluetooth

In this example, we will wirelessly connect a Bluetooth audio source and use the IoT RedBoard ESP32 as a Bluetooth audio sink. Once the audio codec is set as an I^2S peripherial, the ESP32 will receive audio and play it back via I^2S. Once the WM8960 receives the I^2S audio, it will be sent to the DAC and then the headphone output.

Hardware Hookup

Note: For users using the SparkFun Qwiic Wireless Speaker Kit, the kit includes a TRRS breakout with headers instead of the TRS breakout. RING2 of the TRRS connector also connects to the TRS cable's sleeve. When wiring your circuit up for this example, we recommend connecting to the sleeve in case you have a headphone with a TRRS connector.

Connect power, I^2C, I^2S, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



Connect a USB cable into your IoT RedBoard ESP32. Turn on the Bluetooth on your audio source (e.g. MP3 player, smartphone, or computer). Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_09_I2S_Bluetooth**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

Open the Arduino Serial Monitor and set it to **1152000** baud to view the serial output. On your phone, pair and connect to the ESP32's Bluetooth. In this case, the name of the ESP32's Bluetooth is called "**myCodec**". Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones.

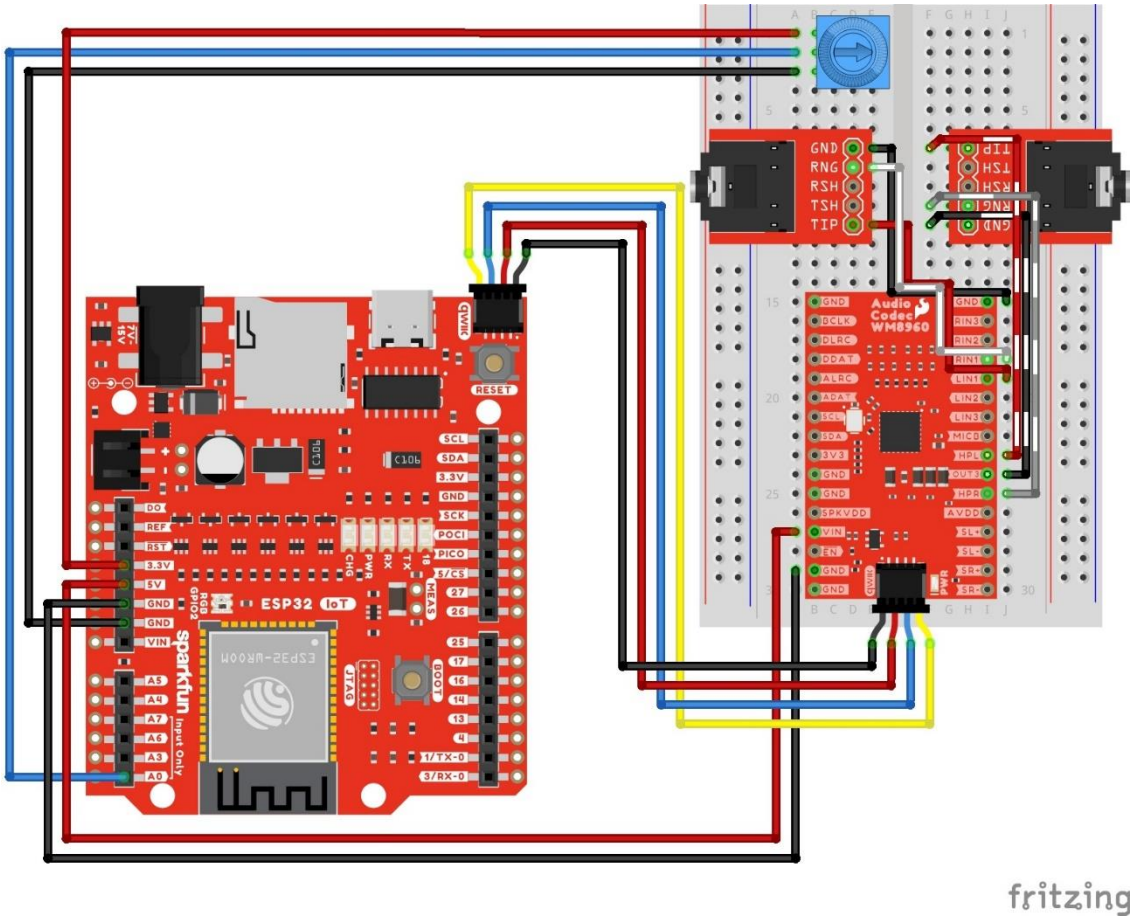
Try adding a battery to the mix to untether the circuit from your computer. Do a little dance or flip and rock to the beat of the music! Of course, you'll want to be careful of the wires while moving around. For a more secure connection, you could take a solderable breadboard or an Arduino shield and manually wire the circuit together

Example 10: ADC Gain

This example is pretty much example 5. The difference is that we are adding a trim pot to the ESP32's analog input and using the measurement to adjust the codec's ADC digital volume.

Hardware Hookup.

Connect power, I^2C, line input 1, trim pot, and the headphone output. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_10_AdcGain**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

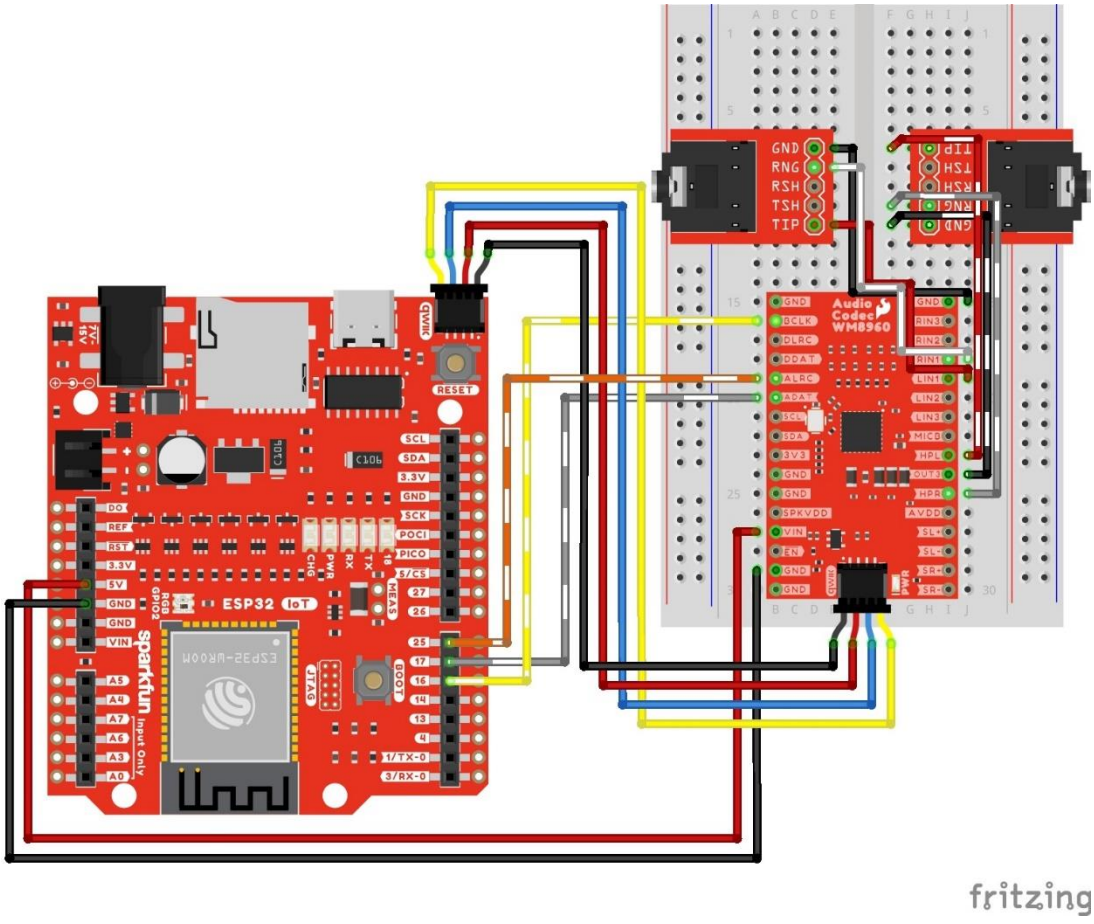
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones. Adjust the volume to your desired setting by rotating the trim pot clockwise or counterclockwise.

Example 11: Volume Plotter

This example is similar to example 3. However, we will be connecting to reading I2S audio from the ADC and plotting the audio samples on the Arduino Serial Plotter.

Hardware Hookup

Connect power, I^2C, I^2S, line input 1, and the headphone output. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_11_VolumePlotter**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

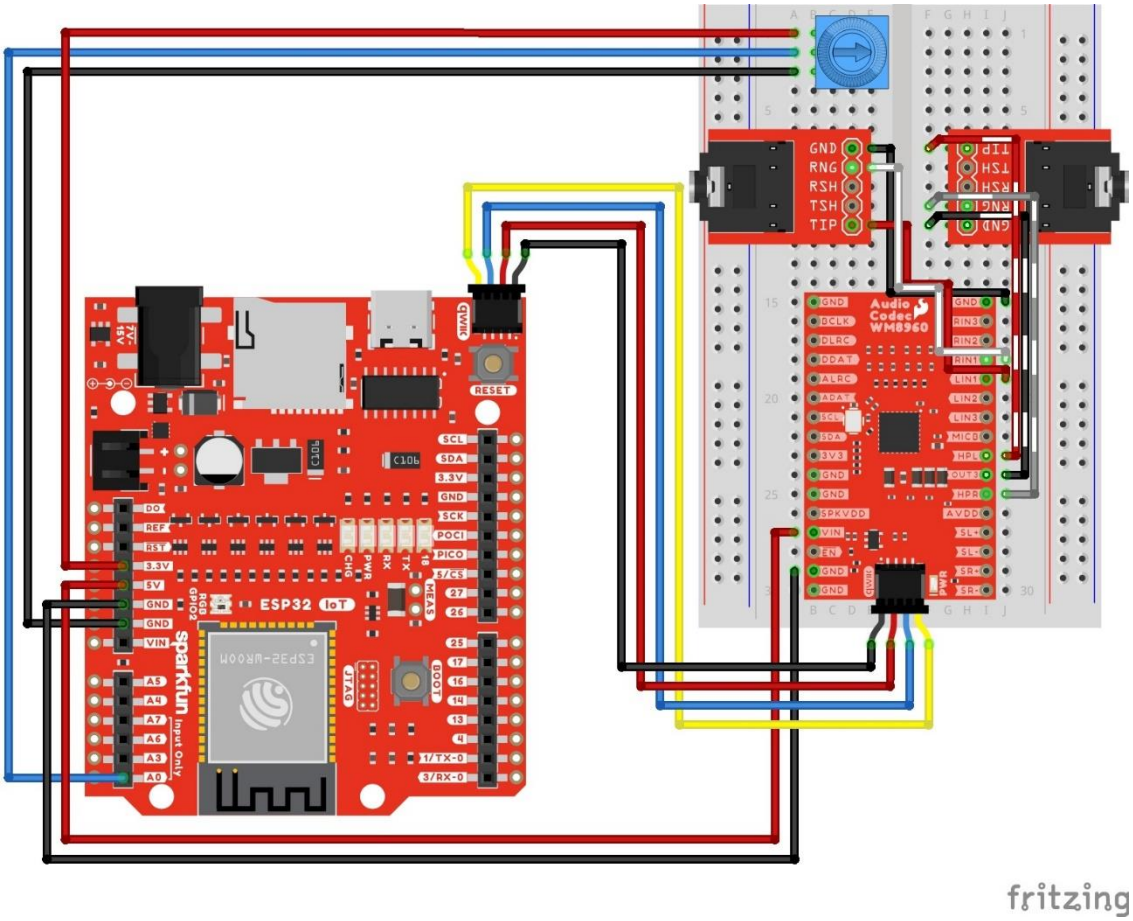
Open the Arduino Serial Plotter and set it to **115200** baud to view the output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones and view the audio samples on the Serial Plotter.

Example 12: Automatic Level Control

This example builds off of example 5. The difference is that we are adding a trim pot toe the ESP32's analog input and using the measurement to configures the Automatic Level Control (ALC) target value. The ALC will adjust the gain of the PGA input buffer to try and keep the signal level at the target.

Hardware Hookup

Connect power, I^2C, line input 1, trim pot, and the headphone output as explained earlier. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_12_AutomaticLevelControl**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

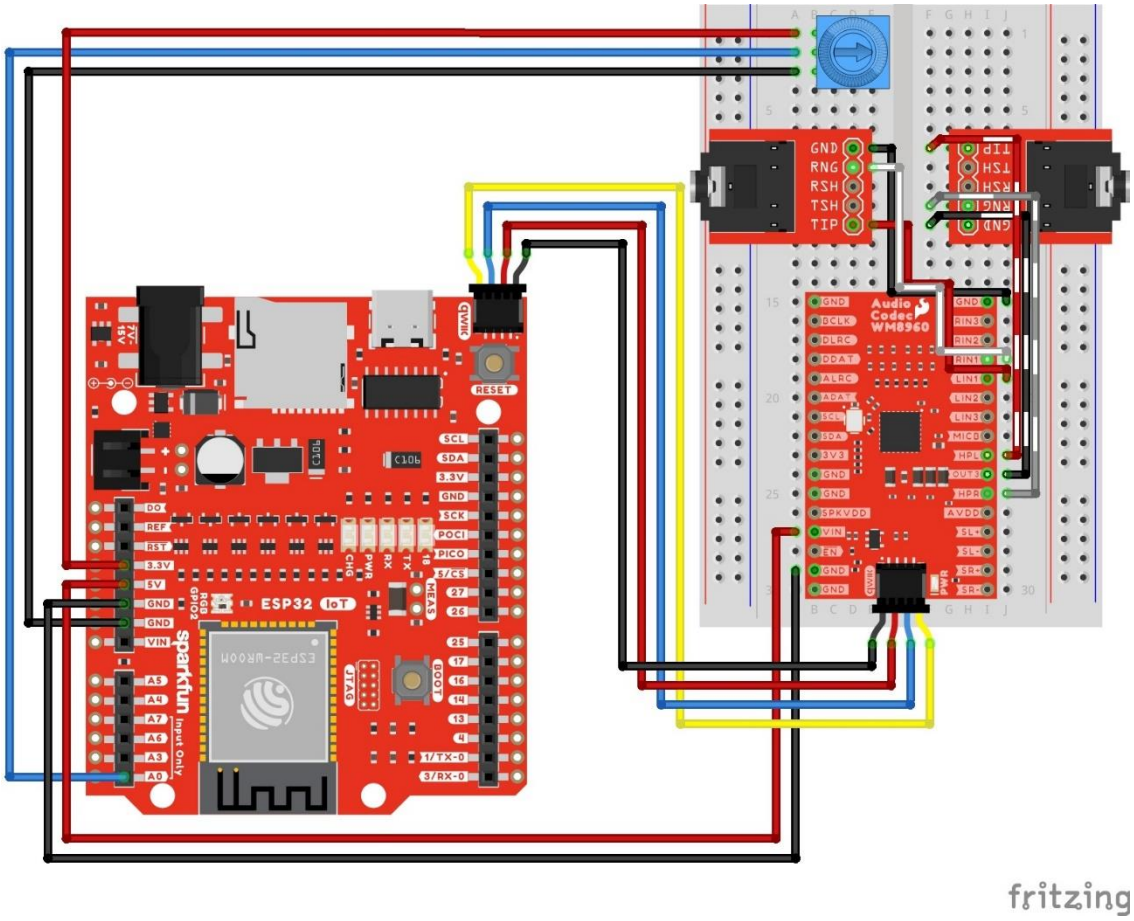
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones.

Example 13: DAC Gain

This example is similar to example 10. The difference is that we are adding a trim pot to the ESP32's analog input and using the measurement to adjust the codec's DAC digital volume.

Hardware Hookup

Connect power, I^{2}C, I^{2}S, line input 1, trim pot, and the headphone output. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32 and a TRS cable from your audio source (e.g. MP3 player, smartphone, or computer) into the audio connector. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_13_DacGain**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

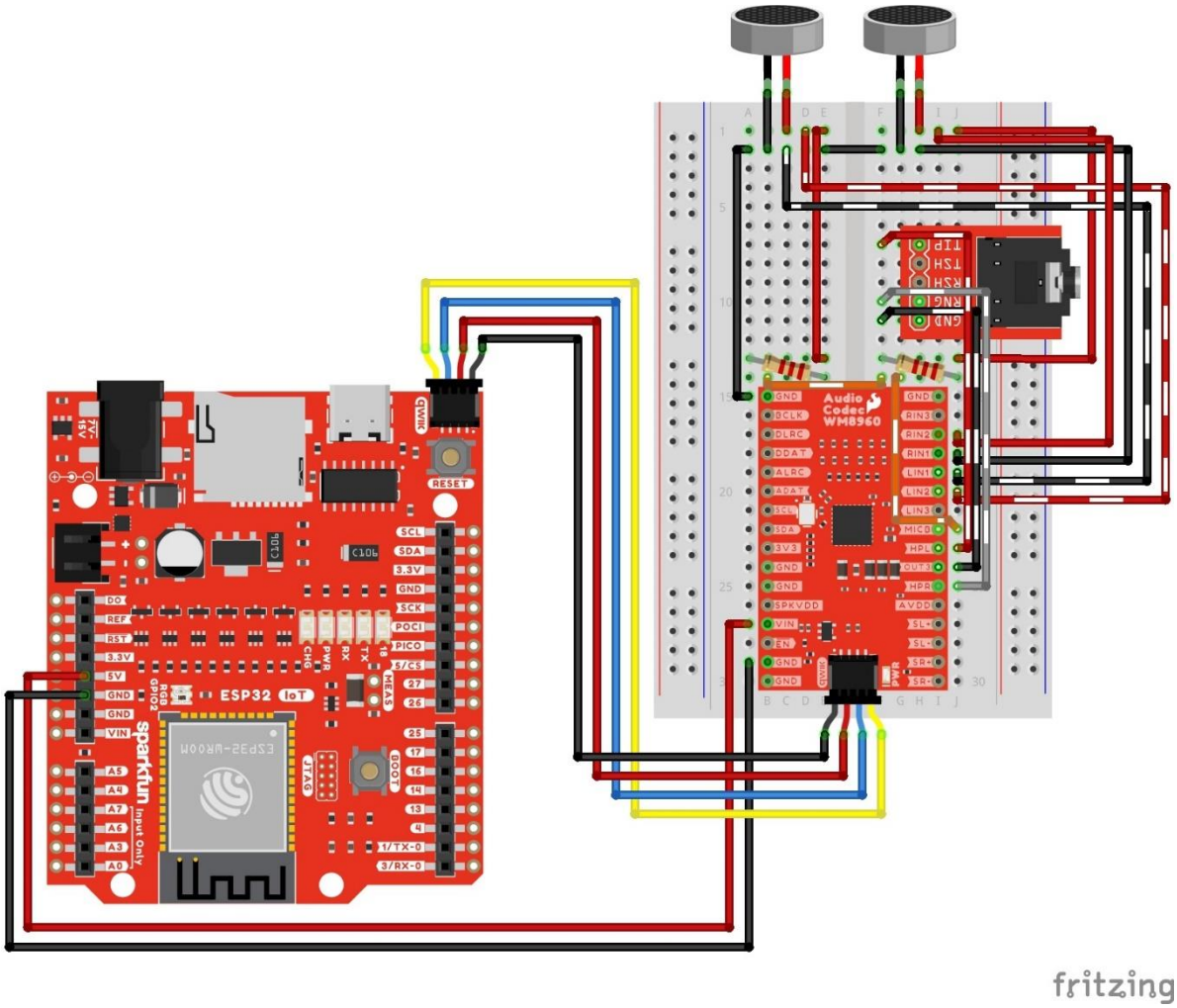
Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Then hit the play button on your audio source. Make sure that the volume from the audio source is turned up. You should hear some music from the headphones. Adjust the volume to your desired setting by rotating the trim pot clockwise or counterclockwise.

Example 14: Electret Mic

Continuing on from example 7, we will use differential microphones as the audio input based on the "pseudo-differential MIC configuration." The example will configure the PGA before passing it to the mixers and gain stages of the codec like the previous examples. The output will be to the headphones.

Hardware Hookup

Connect power, I^2C, I^2S, differential microphones (each with a 2.2kΩ resistor wired in series with the MICBIAS pin), and the headphone output. Your circuit should look similar to the circuit diagram below.



If you have not already, insert a USB cable into your IoT RedBoard ESP32. Then connect your headphones to the output. For those that are sensitive to sounds, you may want to hear the example output before inserting the headphones into your ears.

Upload Code

From the menu, select the following: **File** > **Examples** > **SparkFun WM8960 Arduino Library** > **Example_14_Electret_Mics**. If you have not already, select your Board (in this case the **SparkFun ESP32 IoT RedBoard**), and associated COM port. Then hit the upload button.

Open the Arduino Serial Monitor and set it to **115200** baud to view the serial output. Make some noise on the differential microphones. You should hear some audio from the headphones.

Troubleshooting

Not working as expected and need help?

If you need technical assistance and more information on a product that is not working as you expected, we recommend heading on over to the SparkFun Technical Assistance page for some initial troubleshooting.

If you don't find what you need there, the SparkFun Forums are a great place to find and ask for help. If this is your first visit, you'll need to create a Forum Account to search product forums and post questions.

Resources and Going Further

Now that you've successfully got your Audio Codec Breakout up and running, it's time to incorporate it into your own project! For more information, check out the resources below:

- Schematic (PDF)
- Eagle Files (ZIP)
- Board Dimensions (PNG)
- Fritzing Part
- WM8960 Datasheet (PDF)
- Arduino Library
- GitHub Hardware Repo
- SFE Product Showcase

Looking for more inspiration? Check out these other tutorials related to audio.