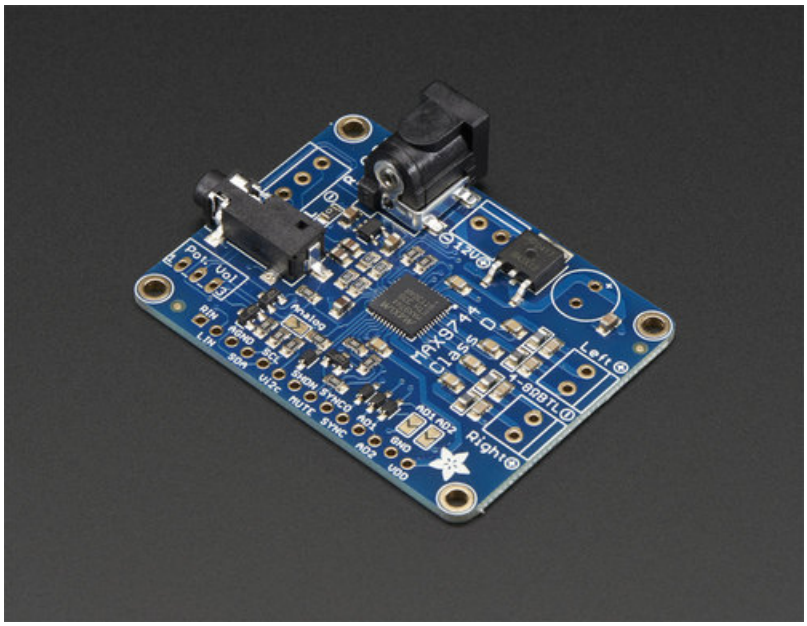
adafruit learning system

Adafruit 20W Stereo Audio Amplifier - MAX9744

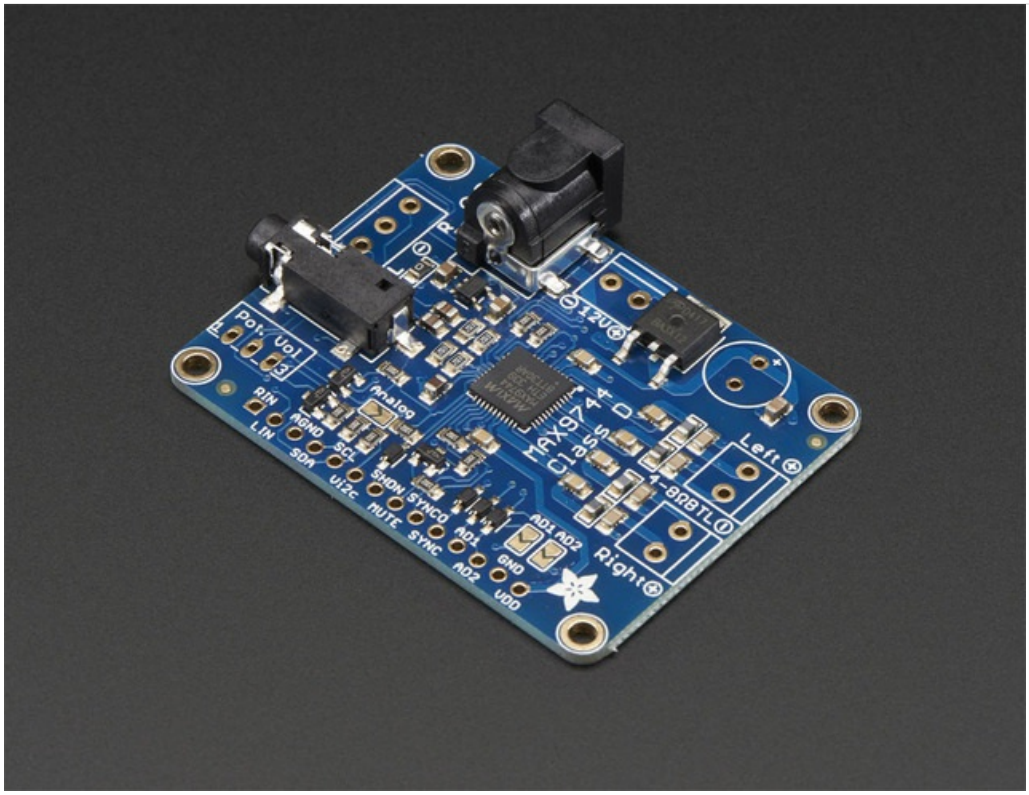
Created by lady ada



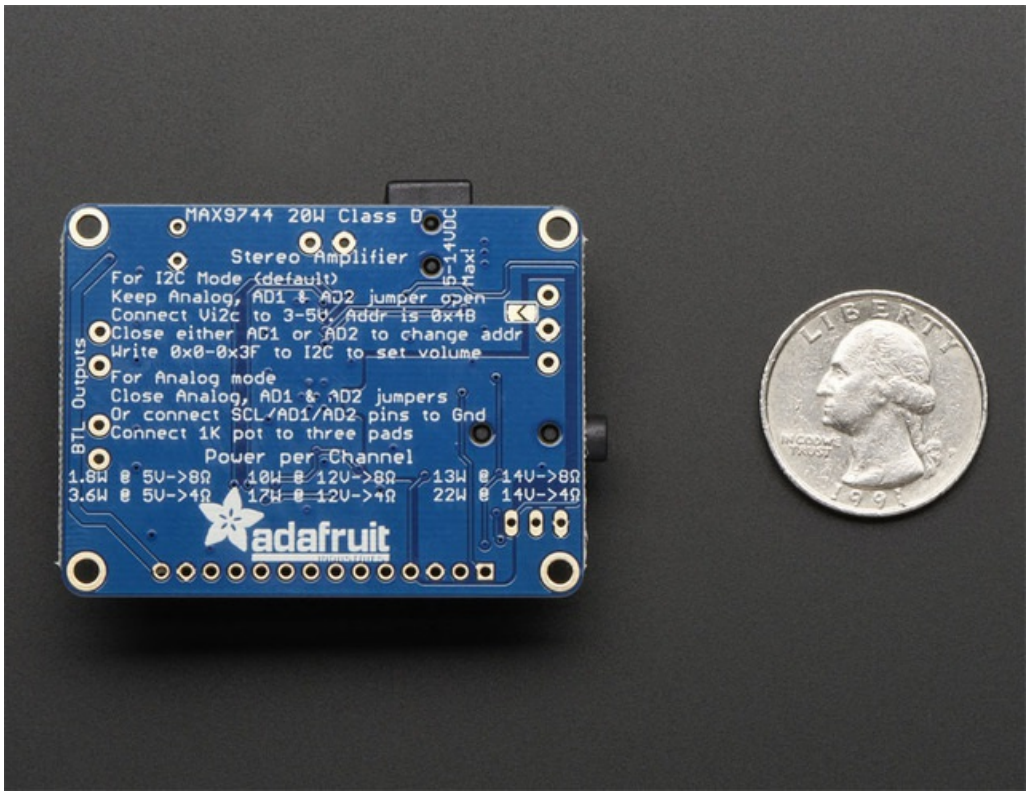
Last updated on 2018-03-05 11:06:49 PM UTC

Guide Contents

Overview



Pump up the volume with this 20W stereo amplifier! This slim little board has a class D amplifier onboard that can drive 2 channels of 4-8 ohm impedance speakers at 20W each. Power it with 5-12VDC using the onboard DC power jack and plug stereo line level into the 3.5mm stereo headphone jack and jam out with ease. Since it's class D, its completely cool-running, no heat sinks are required and it's extremely efficient - up to 93% efficiency makes it great for portable or battery powered rigs.

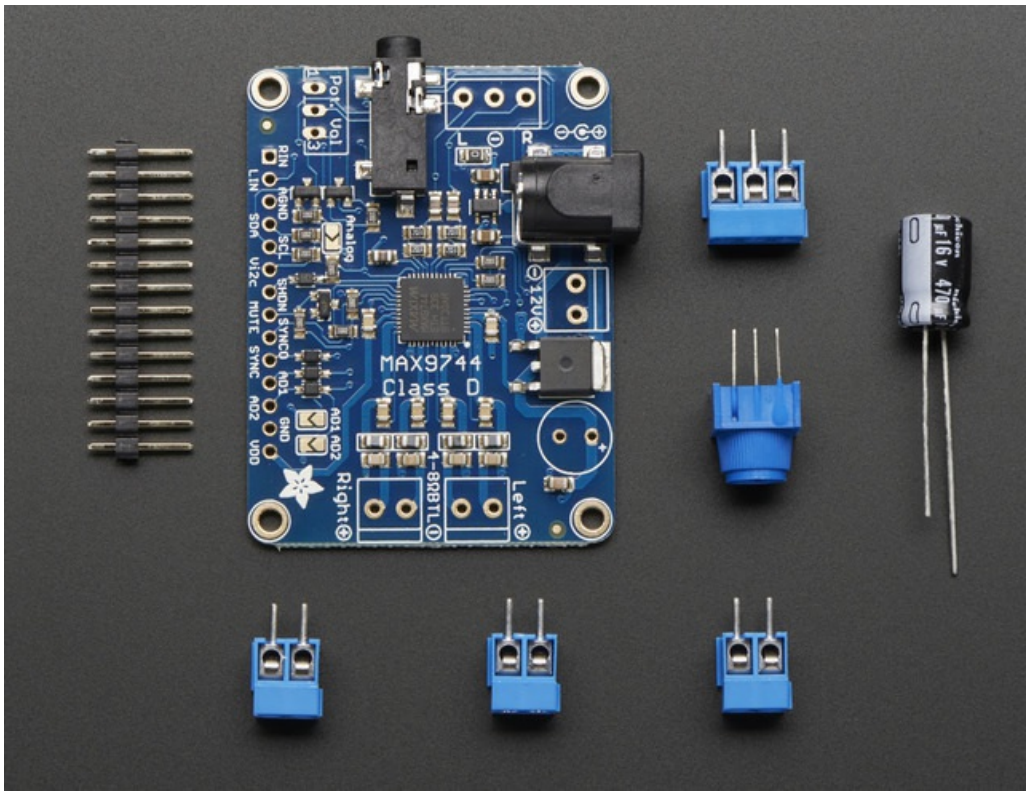


We like the MAX9744 amplifier at the heart of this board because its very easy to use, but it also has *both analog and digital* volume control capability. Use a single 1KΩ pot (we include one) to adjust volume analog-style. Or hook it up to your favorite microcontroller and send I2C commands to set 64-steps of volume amplification.

Some great stats about the MAX9744

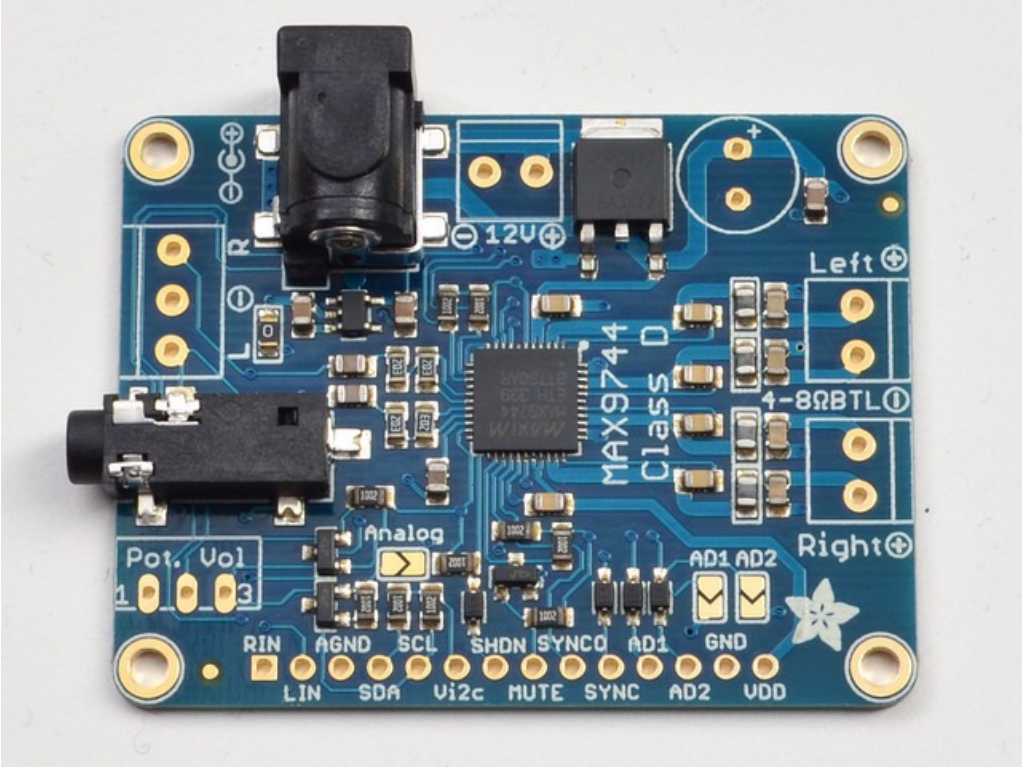
- Power from 4.5V-14V DC voltage
- Up to 93% efficient (88-93% typical)
- 20mA quiescent current (or put into shutdown for 1uA quiescent)
- Up to 29.5dB max gain
- Use DC or AC coupled line-level input, up to 3Vpp
- Filterless Spread-Spectrum Modulation Lowers Radiated RF Emissions from Speaker Cables
- 20W Stereo Output (4Ω, VDD = 12V, THD+N = 10%)
- Low 0.04% THD+N
- Integrated Click-and-Pop Suppression
- Short-Circuit and Thermal-Overload Protection

We took this lovely chip and wrapped it up into a breakout for you, with polarity-protection, jacks and terminal blocks, i2c level shifting, and a spot to solder in a volume pot.



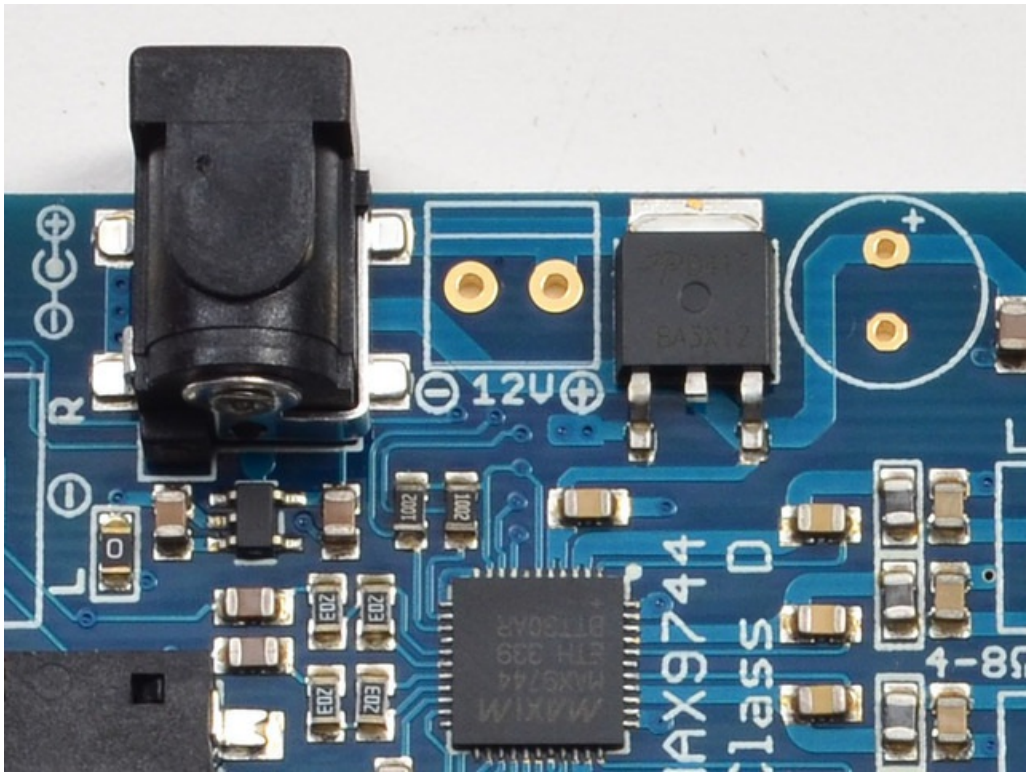
Each order comes with one MAX9744 breakout board with all surface-mount parts fully assembled and tested. We also include 3 x 2pin and 1 x 3pin terminal blocks, a 470uF power filter capacitor and 1KΩ trim pot. To use this board, a little soldering is required to attach the terminal blocks and other components, but its fairly easy and expect it should take less than 15 minutes.

Pinouts



Power connections

The MAX9744 amplifier can use between 5-14VDC power. The higher the voltage, the more gain you can get. So if you want 20W per channel, you'll need to supply 12VDC. The amplifier is a Class D - so it only draws current when its playing audio, but the voltage requirement is still pretty important. If you pick a voltage too low, you'll hear distortion on the speakers because the output is 'clipping'



There's two ways to get power into this board, either by the 2.1mm DC jack (on the left) or the 3.5mm terminal block breakout (in the middle). Both are connected up together so use whichever you like.

The 2.1mm DC jack is a 'standard' 2.1mm/5.5mm barrel jack, with center positive connection. The terminal block has markings showing which pin is positive and which is negative. If powering from a wall adapter, use the DC Jack, the terminal block is best for battery packs with wires coming out of them. You'll need to solder the terminal block into this spot, see the Assembly page for details

To the right of the terminal block is a polarity protection MOSFET, it will make sure you only provide positive voltages to the board! If the polarity is negative, it just wont work. Its more likely to happen with the terminal blocks but its always nice to have a protection circuit.

All the way on the right is a spot for a electrolytic capacitor. We include a 470uF capacitor in the kit. This capacitor helps smooth out the power supply. This isn't required when running off of batteries, but if you're using a wall adapter, especially a really old 'transformer' style one, this will be required. If you want the capacitor, solder it in using the instructions in the Assembly section

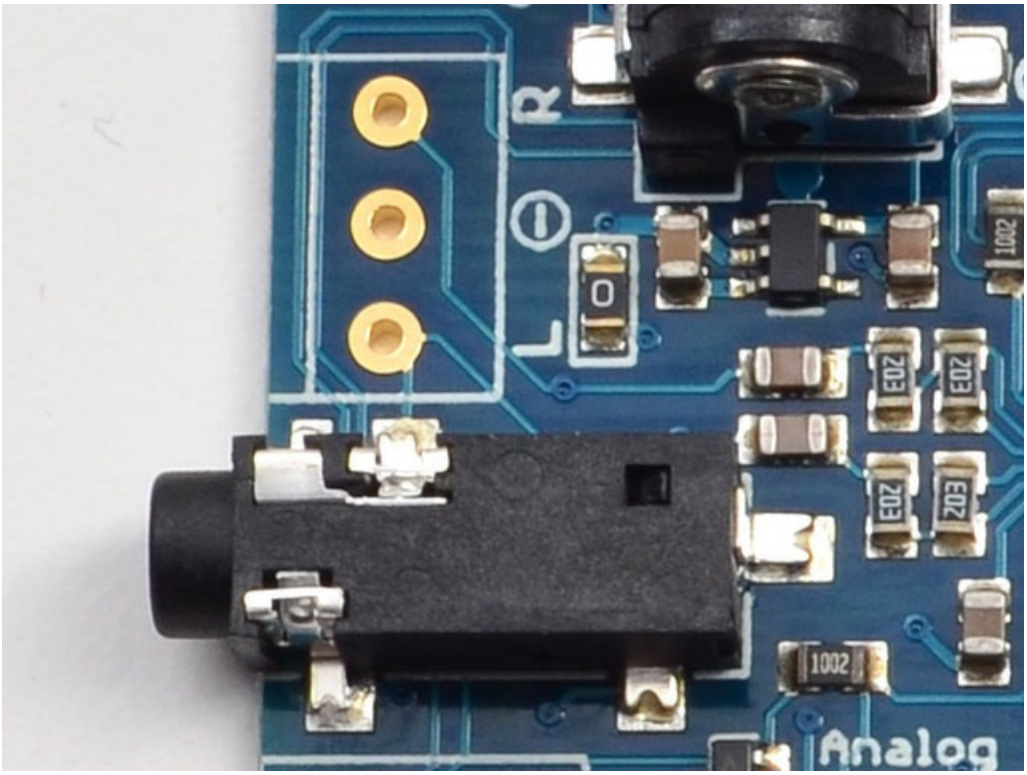
The MAX9744 board draws about 30-50mA quiescent current - that means that even when not amplifying audio you'll be drawing that much current from the power supply. On top of that, you have to add the current usage for audio amplification which will vary widely based on audio volume, speaker impedance, amplification, etc.

Audio Inputs

This audio amplifier takes in stereo audio, either using a 3.5mm stereo jack or terminal blocks. Line in audio is a-OK. **The audio inputs are not differential!** The ground connection is connected directly to the power ground, this chip simply does not handle differential inputs. However, the inputs do have blocking capacitors, so if your audio levels have DC bias, its OK to connect them up directly without extra audio blocking caps.

Line level audio (about 1Vpp) is suggested, but it can handle up to ~3Vpp inputs.

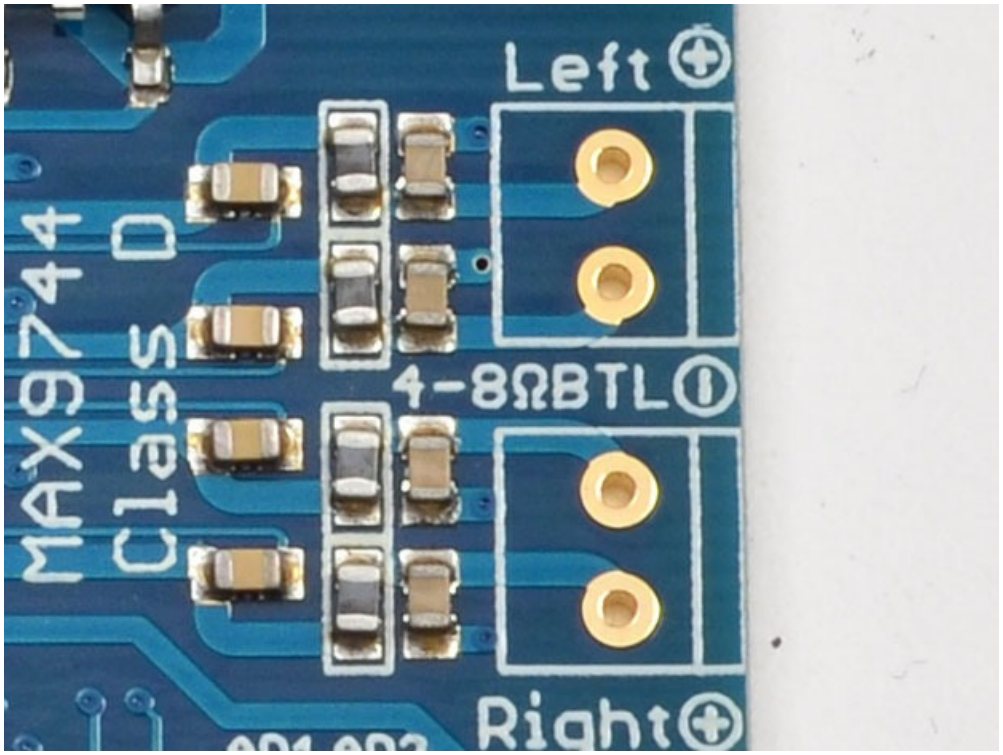
Oops, on the first rev of the PCB we made a mistake with the breakout labeling, swapping the two channels. The top pin is LEFT and the bottom pin is RIGHT. The middle pin is still GND



The headphone jack is a classic 3.5mm as seen in just about every audio device. You can connect this directly to any audio output. The terminal blocks are for if you have some direct-wiring project and you want a more permanent connection. You'll have to solder in the terminal blocks if you want to use that technique, see the Assembly page

Speaker outputs

You got this amp to amplify, and this is where you get that amplified signal out! We use two dual 3.5mm terminal blocks for the speakers, Left and Right



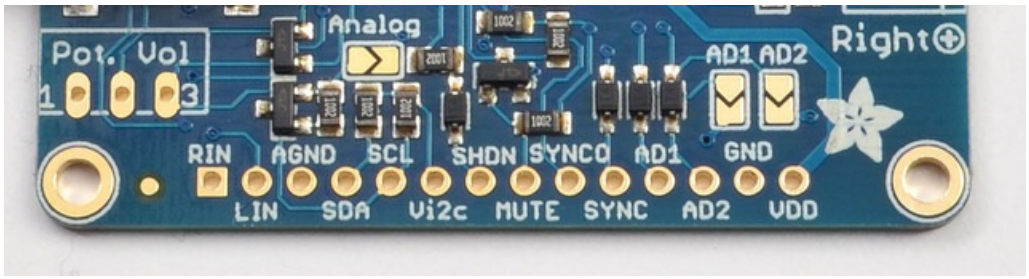
The speakers can be 4 to 8 ohm impedance, 20W maximum power. Since it is a class D amplifier, the signal out of these speaker blocks is a high frequency (~300KHz) PWM square wave. The inductance of the speaker smooths out this signal into audio frequencies of 20-20KHz.

The outputs are Bridge-Tied-Load which means you can only connect speakers up directly. Don't connect the outputs to another amplifier! And you can't "parallel" the two BTL outputs for one 40W load either.

To connect up the speakers you'll want to solder in the terminal blocks, see the Assembly section for details

Breakouts

At the very bottom, we break out all the pins you're likely to use with this amplifier. If you're using the amp in **Analog Mode** you don't need to connect to any of these, most likely. If you're using the amp in **Digital** Mode, you may want to connect to some of them. See the tutorials for each section on how to use the board in either mode.



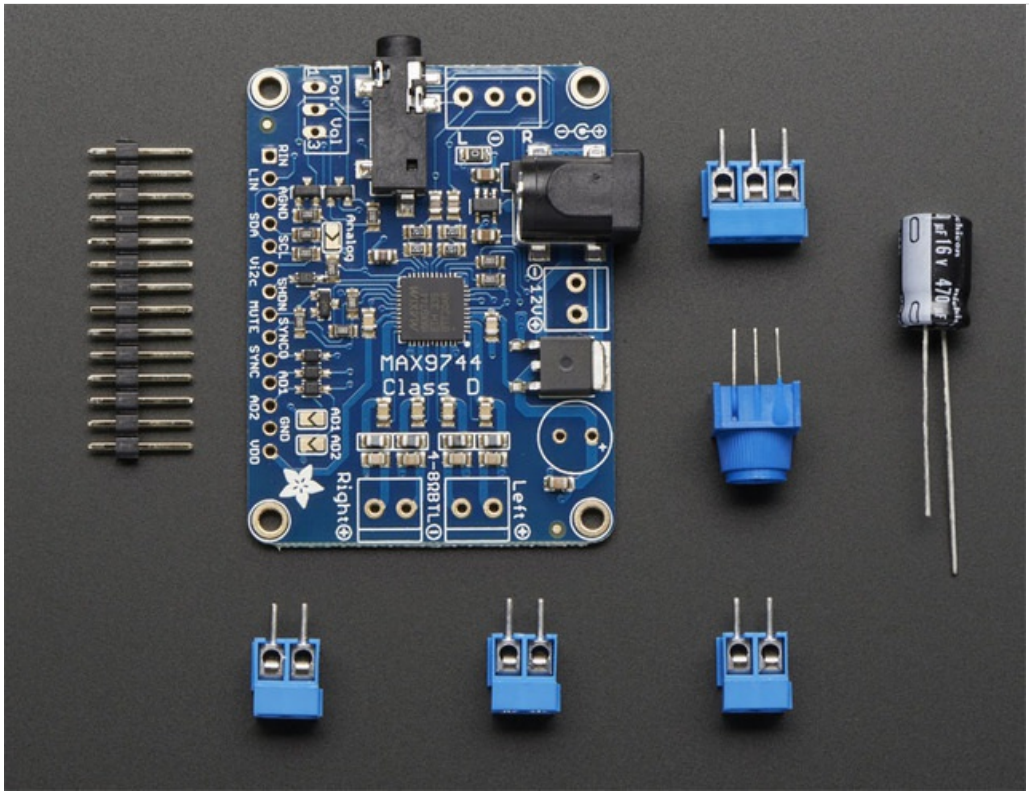
Above the pinouts there are three solder jumpers **(Analog / AD1 / AD2)** and then on the left, a 3-pin breakout called **Pot. Vol** - these are used in analog mode, the solder jumpers are closed to tell the chip we'll be using a potentiometer to set the volume. The **Pot Vol** connection is how we wire up the 1K potentiometer. This is covered in more detail in the Analog Wiring section.

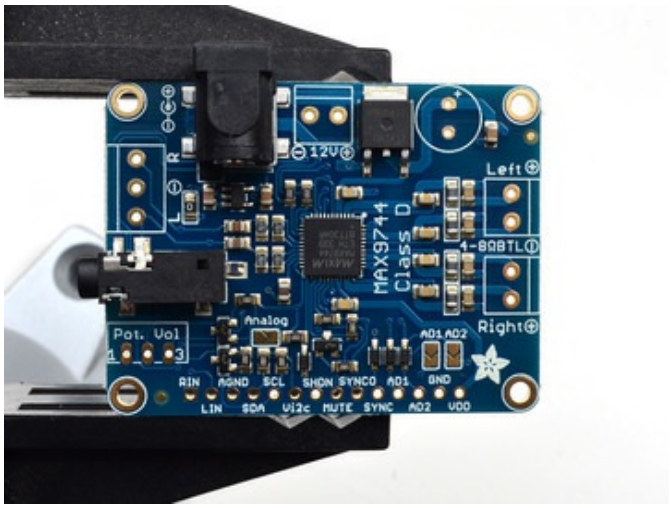
Starting from the left, here are what each pinout connects to:

- **RIN** - this is a duplicate of the audio input terminal block, right channel
- **LIN** - this is a duplicate of the audio input terminal block, left channel
- **AGND** - analog reference ground, the 'quieter' ground for audio signal referencing
- **SDA** - i2c digital signal data, if using Digital Mode to control volume over i2c
- **SCL** - i2c digital signal clock, if using Digital Mode to control volume over i2c
- **Vi2c** - the i2c voltage reference for logic. Only used in Digital mode - connect to 3V or 5V whichever your microcontroller uses
- **SHDN** - digital Shutdown pin. Connect to ground to turn off the entire chip and put it into low power mode
- **MUTE** - digital Mute pin. Connect to ground to turn off only the audio output stages, its faster than shutdown and keeps the digital i2c audio levels
- **SYNCO** - Sync output, this is the high frequency signal from the PWM generator, about 1.4MHz
- SYNC - Sync input, for advanced users who want to clock in their own PWM frequency, keep it above 800KHz.
- **AD1** - I2C address select pin #1
- **AD2** - I2C address select pin #2
- **GND** - Power ground
- **VDD** - 5-12VDC power, from the terminal block/DC jack *after* the polarity protection. You can use this to power your other projects that can handle 5-12VDC power input.

The solder jumper on the back allows you to connect **AGND** (analog ground) to **DGND** (digital ground)

Assembly

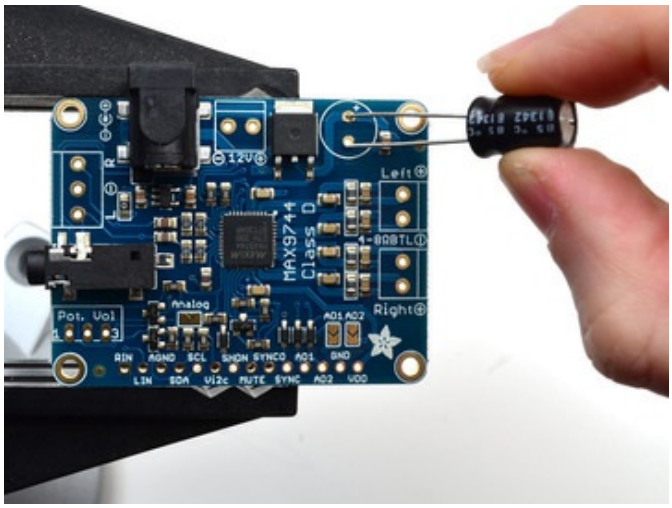




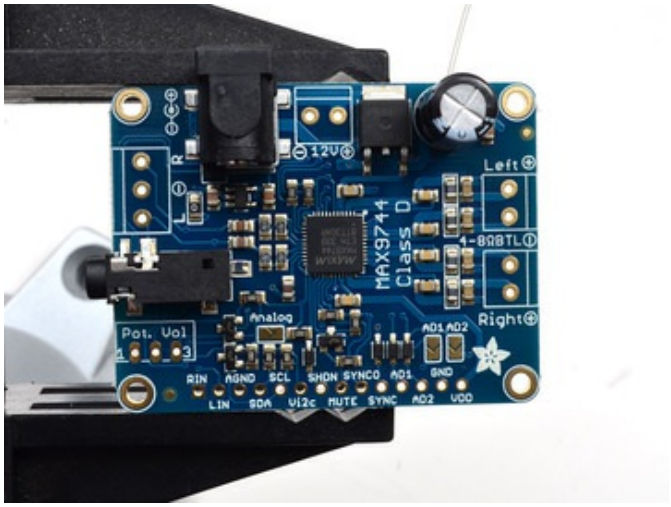
Begin by placing the amplifier board in a vise so you can easily work on it. Heat up your soldering iron to 650-700 degrees F and get some solder and hand tools ready!

Power capacitor

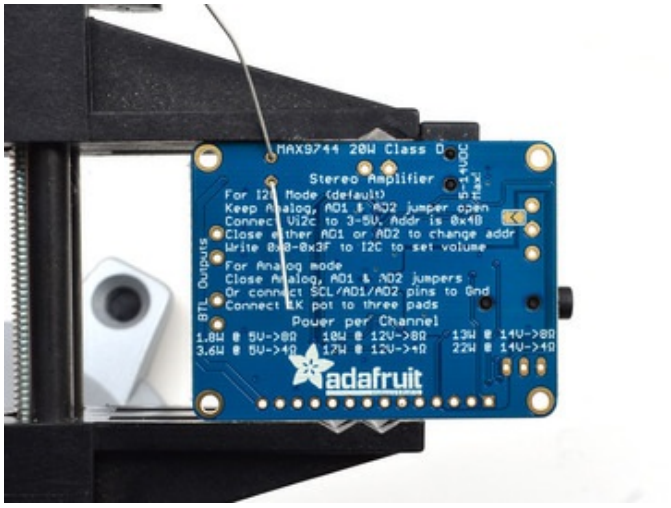
We'll start with the power supply capacitor. This cap isn't required if you're powering off of batteries or a good quality supply, but if you're using wall adapter, this might give you just a little cleaner power especially with high amplification levels



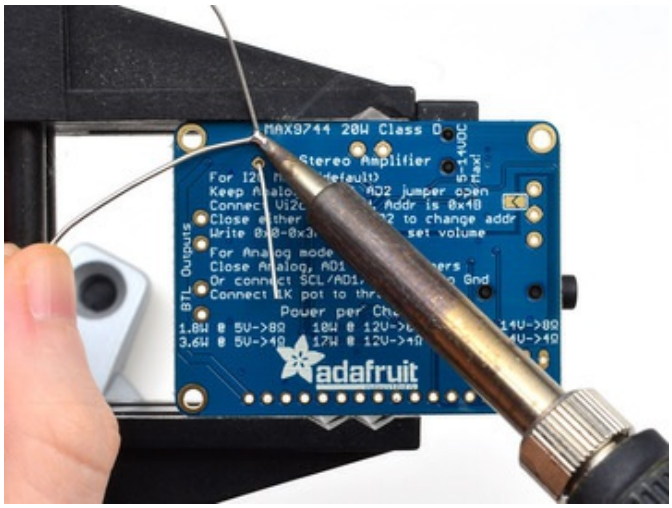
Capacitors are polarized, they have to be placed in the right way!
The longer lead goes into the pad marked +

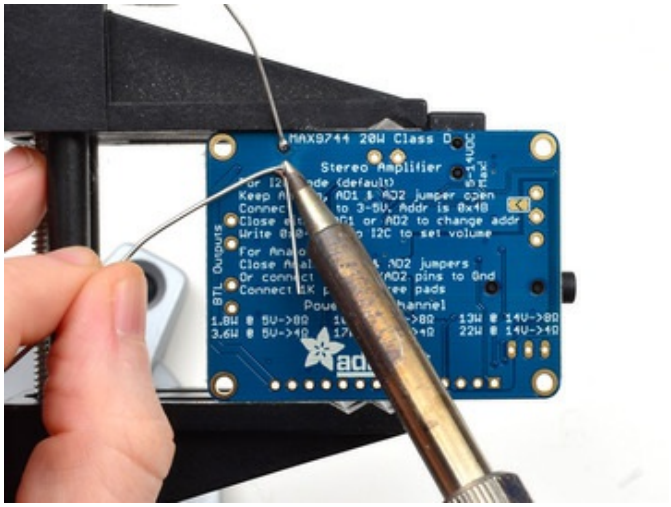


Place the capacitor against the PCB and bend the two leads out so that it sits flat.



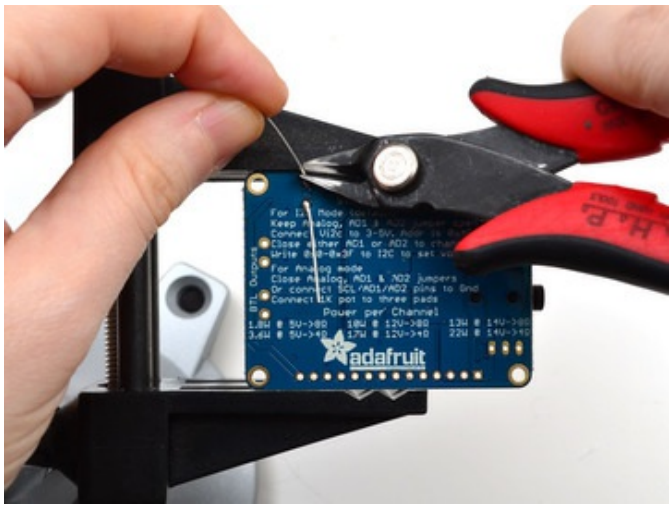
Flip over the board so you can solder the two pads





Solder in both pads, heating the pad and pins at the same time with the edge of the soldering iron and dipping a little solder in.

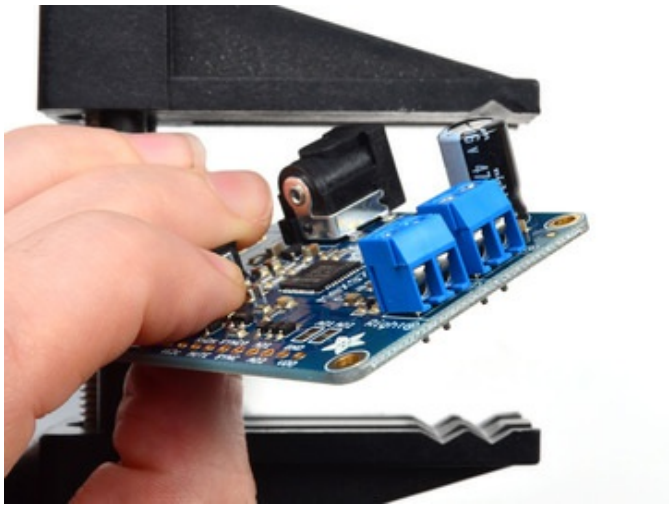
The ground (-) pin may be a little tough to solder since the ground plane acts like a large heat sink



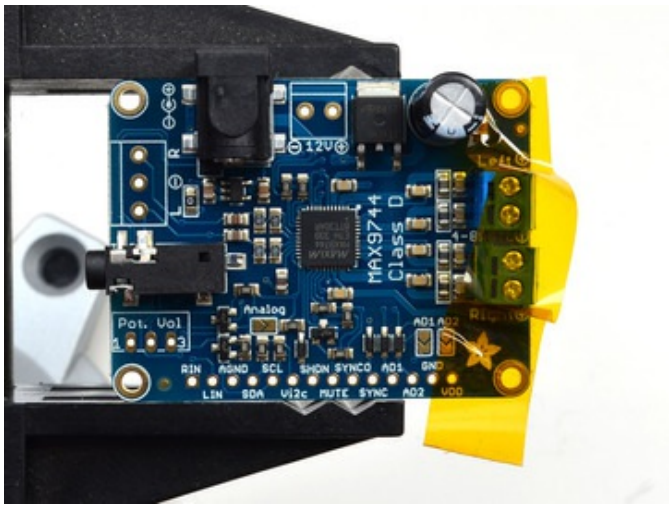
Clip down both of the long leads using diagonal cutters so they are nice and neat

Speaker Terminals

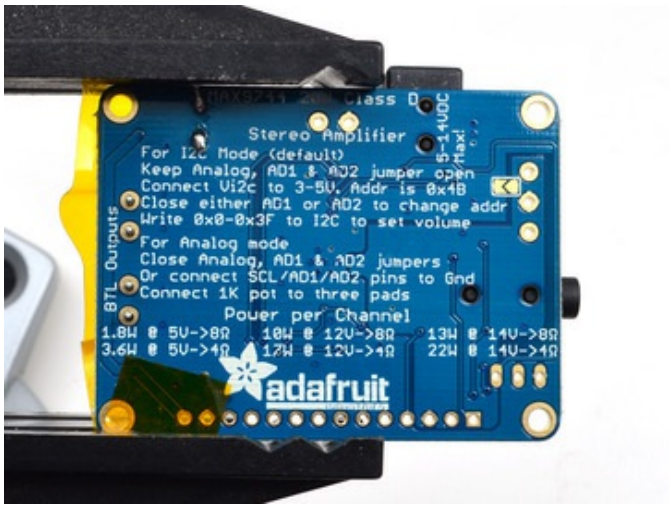
You'll want to do this step, where we add the terminal blocks for the speaker outputs. Otherwise you'd have to solder wires directly to the board which isn't suggested.

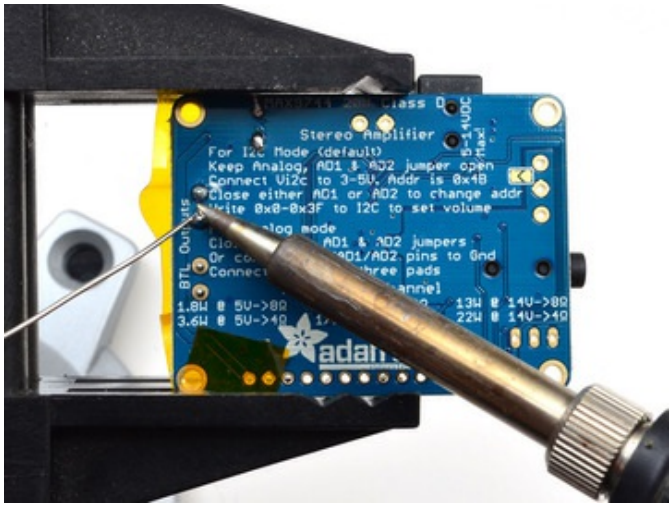


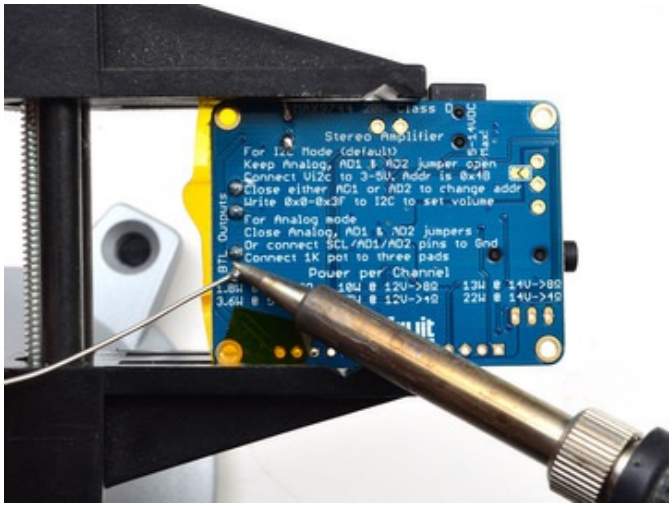
Place two of the 2-pin blue terminal blocks so that the holes point outward



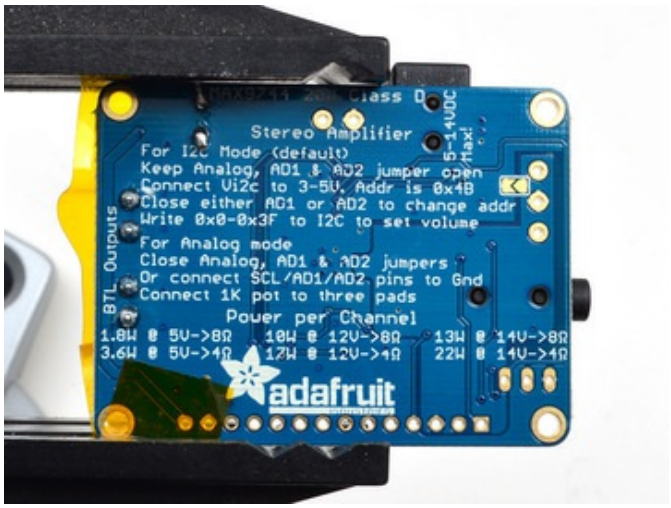
These don't have long leads to bend, so some tape can keep them in place while you solder







Flip over the board again and solder in all 4 pads.

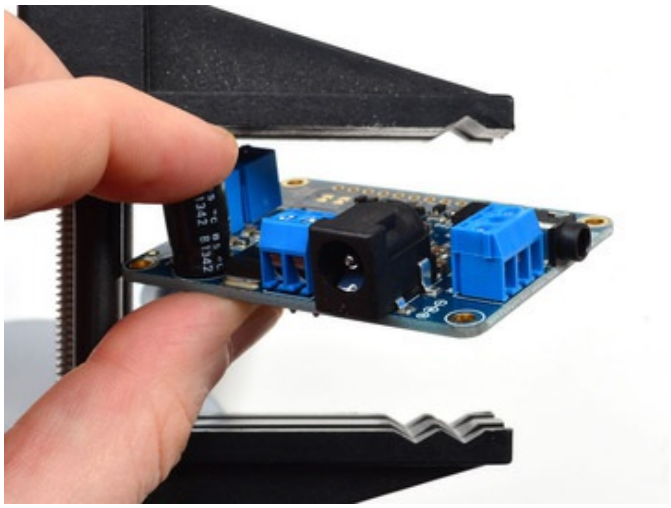


You don't need to clip them because they are already quite short

Remove the tape once you've checked your work

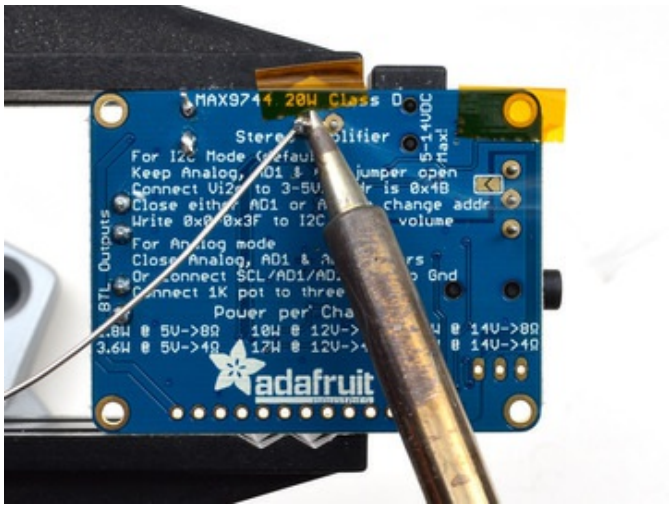
Power and Line In Terminal Blocks

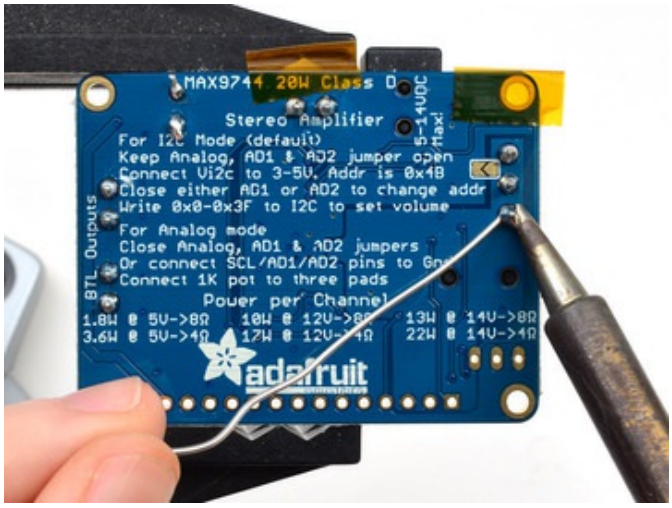
There's also terminal blocks for Power and line-in. These are optional, you can use the DC jack and headphone jack but if you want to hard-wire in, use these instead of soldering directly to the board!



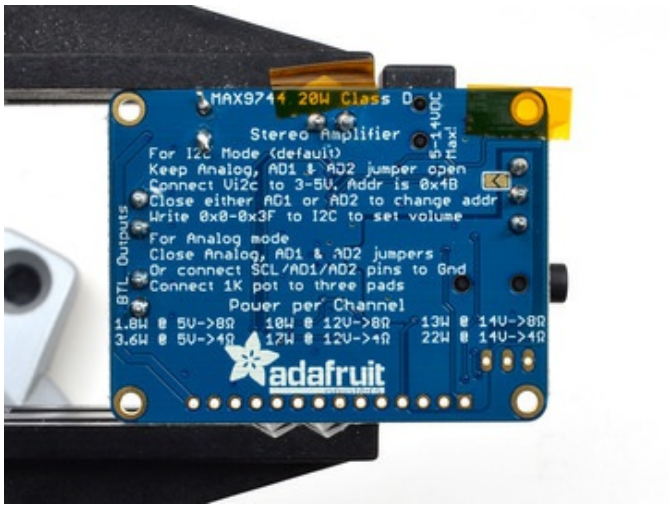
Place the 2-pin and 3-pin terminal blocks so the holes point out

Tape can help here, to keep the blocks in place while you solder.





Solder in all the connections



Check your work before moving onto the Basic Test procedure

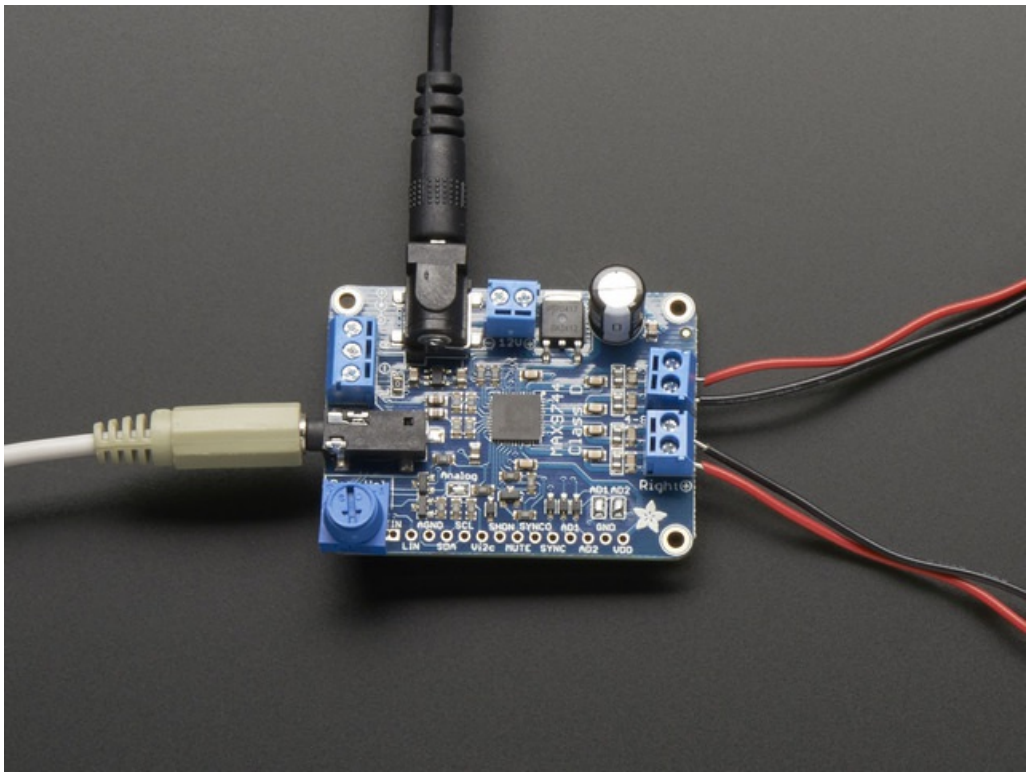
Basic Test

Before you decide whether you want to use Analog control mode or Digital control mode, you should test your amplifier. By default, it will start up with about 6dB of gain, definitely loud enough to be heard!

Begin by connecting two speakers to the BTL speaker outputs of the amplifer

> Don't forget to gently yank on each wire after clamping it into the terminal block, it should be a solid connection, not possible to yank out of place. Loose wires will be a problem!

Next up connect 5-12VDC power to the board using a wall adapter. Then connect up an audio source. I just used a 3.5mm male/male cable to connect it to my audio player.



Starting with low audio volume, slowly turn up the volume on your mp3 player/computer/etc until you hear audio come out. 20W is pretty loud so don't put the speaker next to your ear!

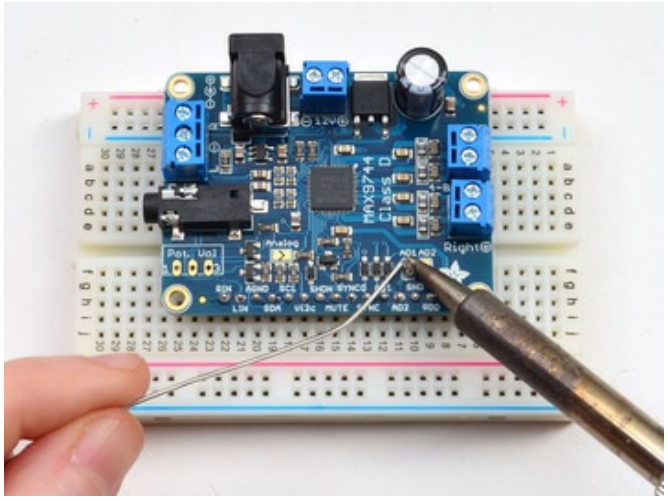
Analog Control

Analog control is the easiest way to get going with the amplifier board. No microcontrollers are required. Just wire up a single 1Kohm potentiometer to select the volume. The potentiometer sets the volume for *both* channels. There is no way to control channels individually.

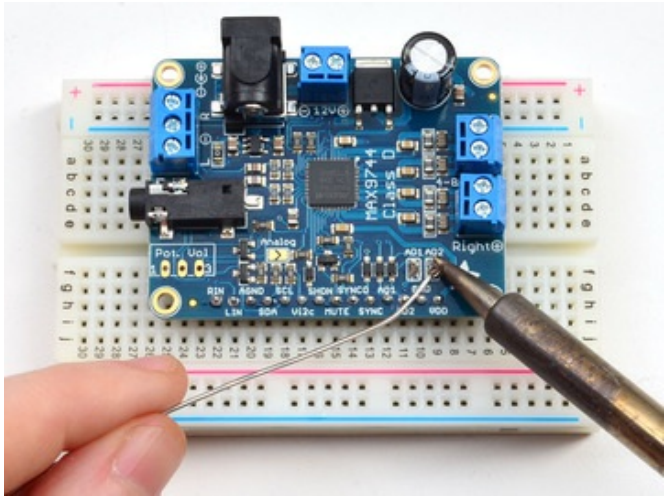
> Before you begin, perform the Basic Test, to make sure that the amplifier is overall working! Then you can add volume control

Preparation

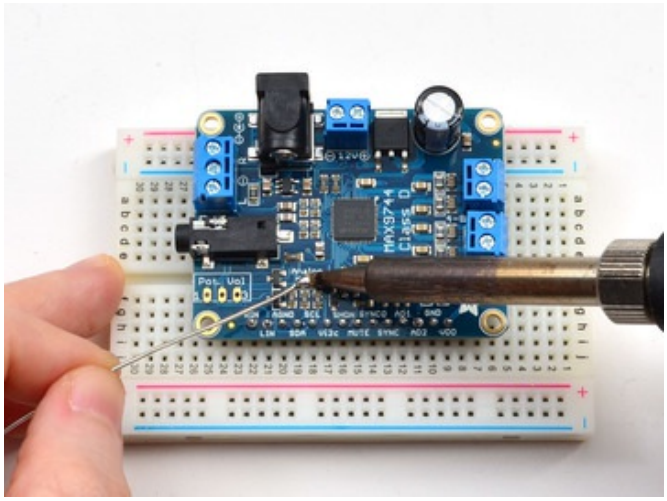
By default, the amplifier breakout is in digital mode. To put it in analog mode we need to close the three solder jumpers labeled **Analog, AD1** and **AD2**.



Solder closed **AD1** on the right side of the board

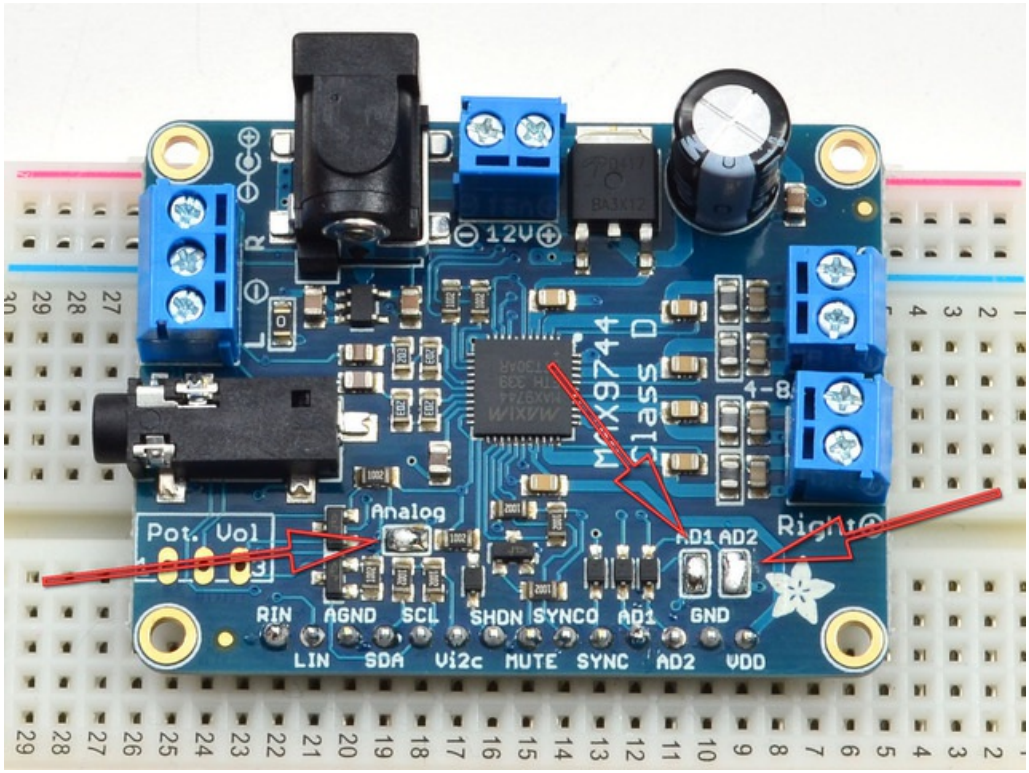


Solder closed **AD2** right next to **AD1**



Finally, solder closed the **Analog** jumper on the left side.

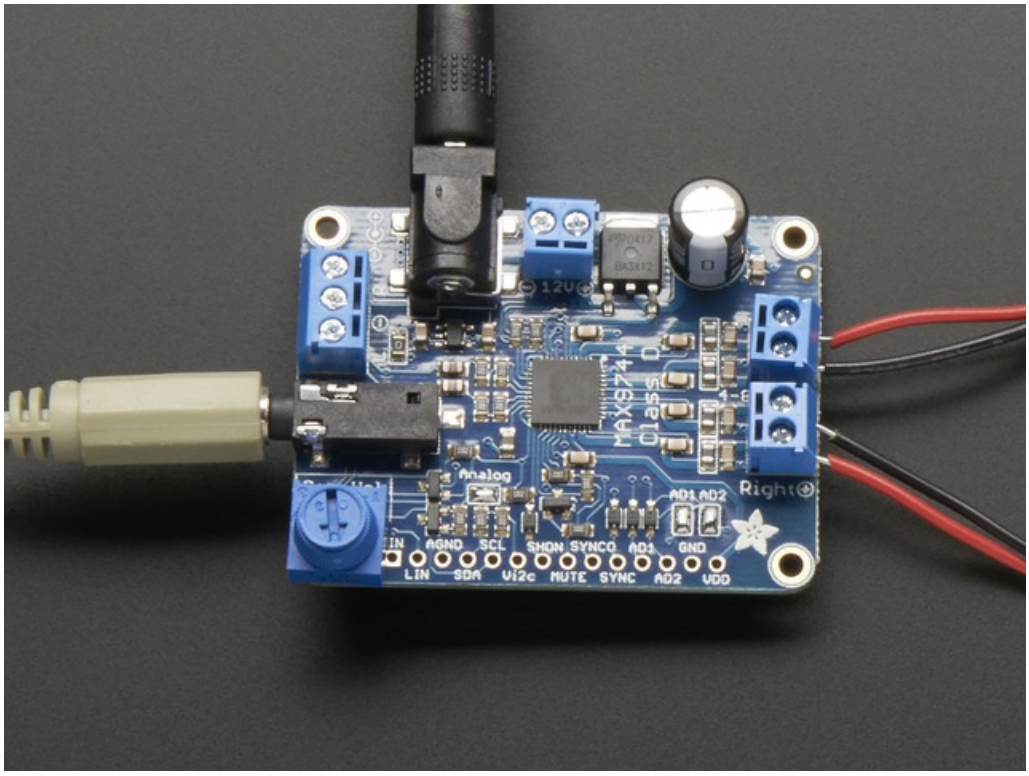
When finished, your amp should look like this, with all three jumpers closed.



When you power up the amplifier and feed in audio **you won't hear anything! This is normal!** If the potentiometer is not installed, it will default to lowest volume.

Now place the potentiometer that came with your kit into the three holes labeled **Pot Vol** - don't solder them in yet, try to just touch the pads to the pins so that you can adjust the volume. As you twist the pot, the volume will go up and

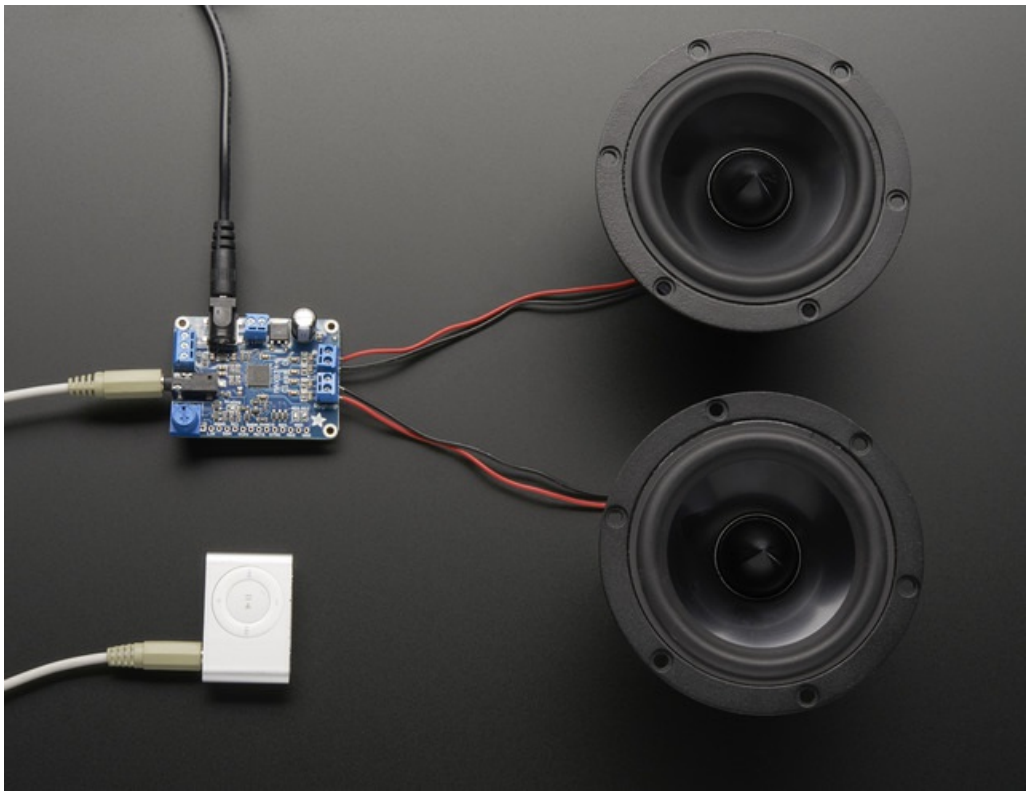
down



Once you have that working, you can solder wires to the **Pot Vol** pads, so you can place the volume adjustment pot elsewhere (say mounted to the outside of a box)

If you want a potentiometer that will fit exactly in the slot, check out this side-adjustment pot http://www.digikey.com/product-detail/en/bourns-inc/3362M-1-103LF/3362M-103LF-ND/1088401

For mounting onto a box or panel check out https://www.adafruit.com/products/562

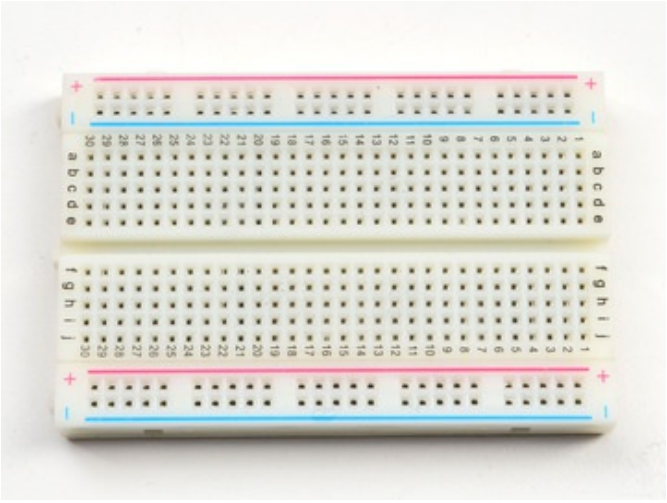


Arduino Digital Control

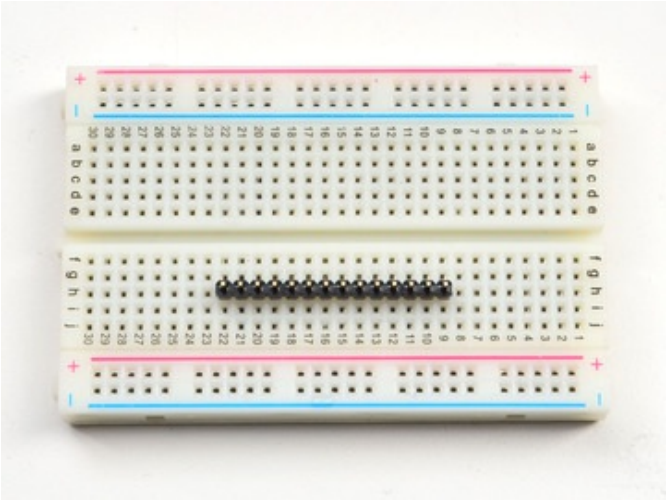
Digital control allows you to set the volume of the amplifier using a microcontroller via I2C protocol. The I2C data pins are connected on a breakout at the bottom of the board.

Assembling Breakout Headers

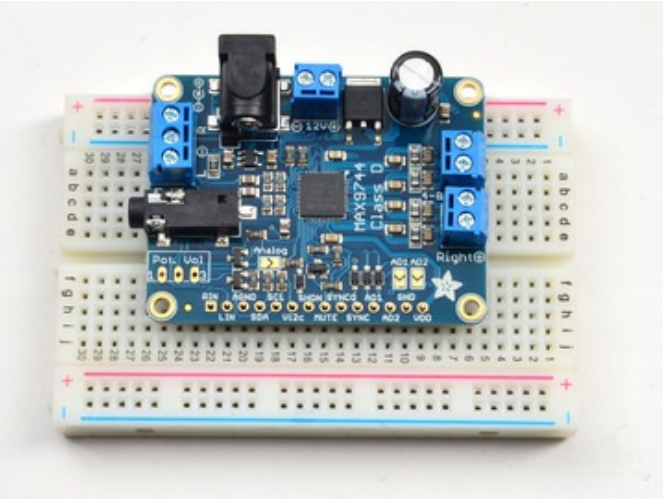
You can solder wires directly to the breakout pads but its easier to prototype if there are headers attached. Follow along to attach headers!



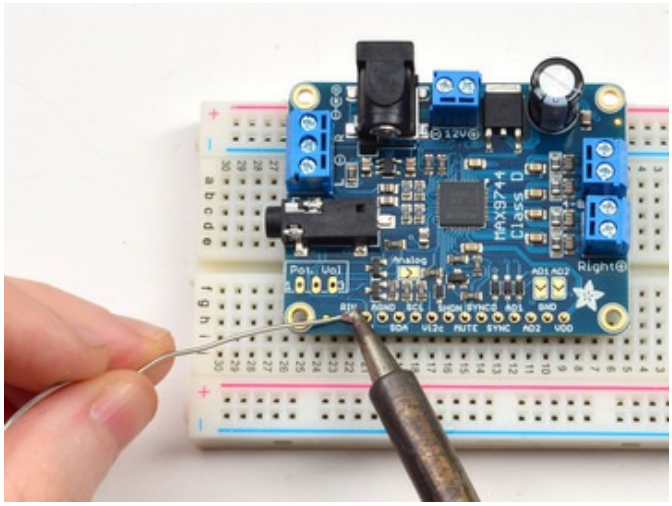
The easiest way to attach headers to the breakout board is to take advantage of a solderless breadboard you may have.



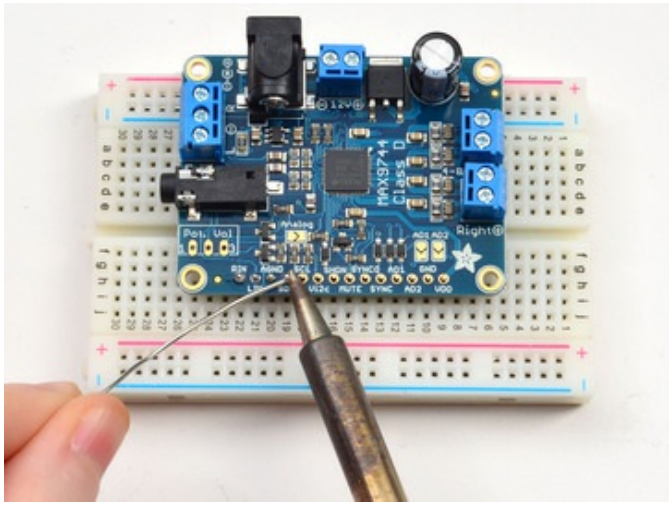
Break off a piece of 0.1" male header, 14 pins long and stick it **long pins down** into the solderless breadboard

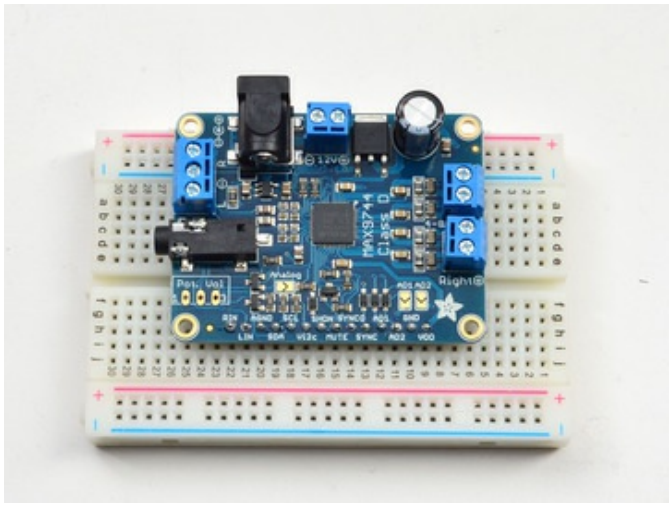


This will keep it in place when you put the audio amplifier on top! Put the breakout board onto the short header pins so they stick out thru the breakout pads



Solder in all of the header pins to the breakout board.





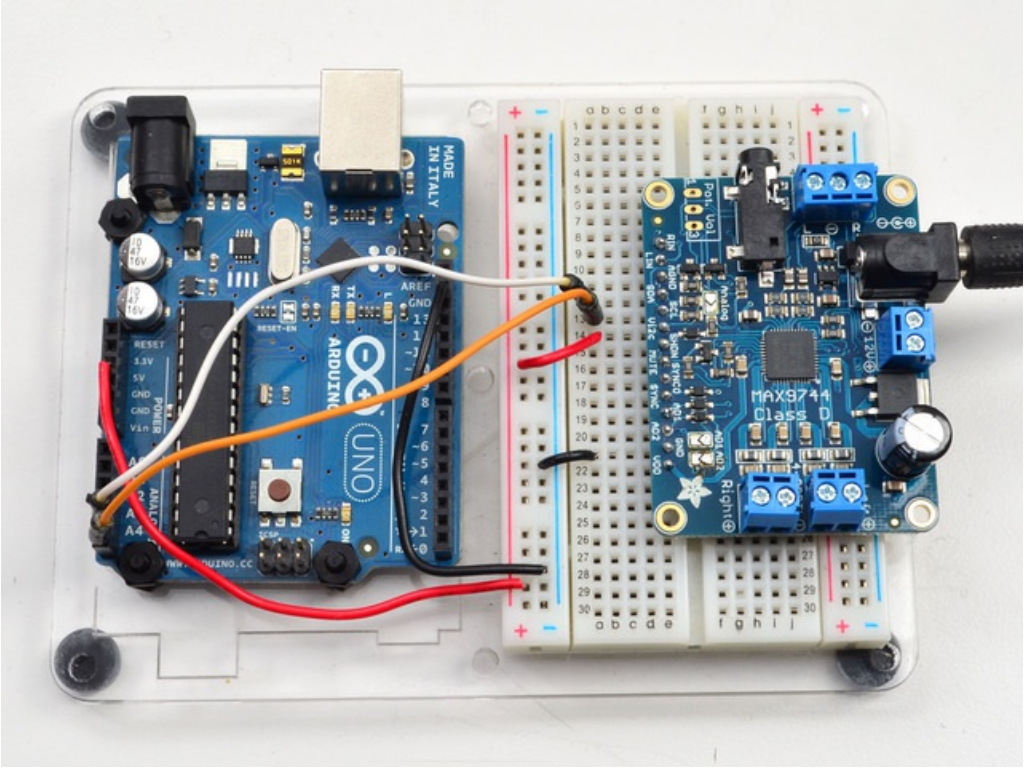
Connecting up to an Arduino

We'll be using an Arduino to test the digital control, but really the I2C protocol is so simple, you can easily port the code to any microcontroller you like.

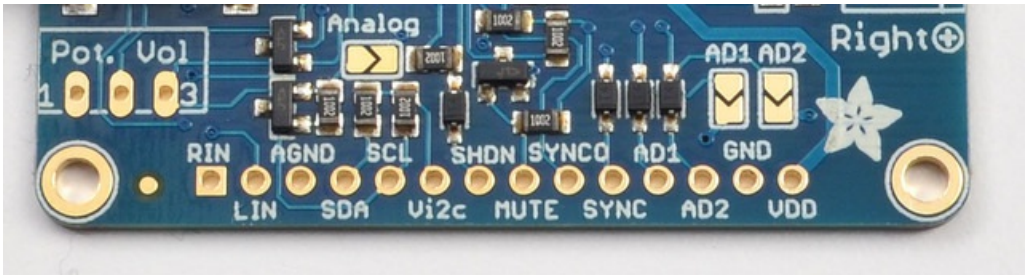
First, though, we'll have to wire up the digital data pins.

Connect the following header pins:

- **GND** from the MAX9744 connects to the common ground connection on your Arduino
- **Vi2c** from the MAX9744 connects to the logic level voltage of your board. For most Arduinos, **5V** works well. If you have a 3V microcontroller, use 3.3V
- Connect the **SDA** pin to the I2C data **SDA** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A4**, on a Mega it is also known as **digital 20** and on a Leonardo/Micro, **digital 2**
- Connect the **SCL** pin to the I2C clock **SCL** pin on your Arduino. On an UNO & '328 based Arduino, this is also known as **A5**, on a Mega it is also known as **digital 21** and on a Leonardo/Micro, **digital 3**



Make sure the AD1 AD2 and Analog solder jumpers are not closed before running the I2C tests, or it won't work! For digital mode, these jumpers must be un-soldered



Run code

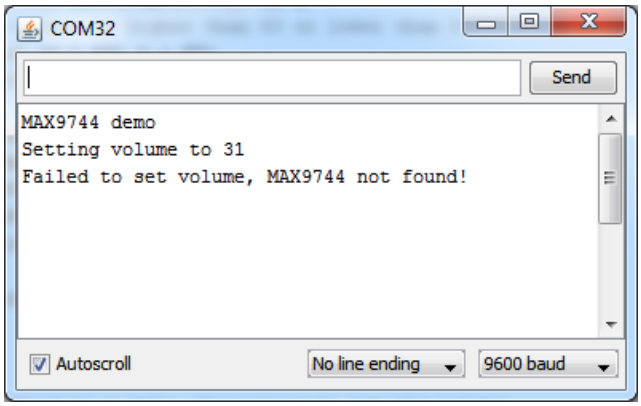
Since the MAX9744 has pretty much no protocol for sending data other than "write the volume level you want" we didn't write a library, just an example sketch as shown below

Upload this sketch to your Arduino, and keep it connected to your computer. Don't forget to power the MAX9744 with 5-12VDC separately via the DC jack!

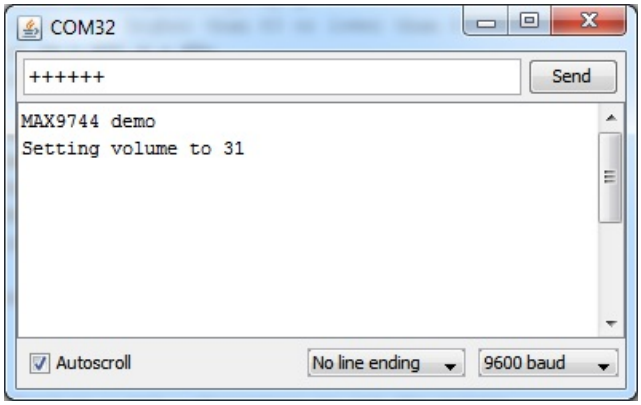
```
#include <Wire.h>

// 0x4B is the default i2c address
#define MAX9744_I2CADDR 0x4B

// We'll track the volume level in this variable.
int8_t thevol = 31;

void setup() {
  Serial.begin(9600);
  Serial.println("MAX9744 demo");
  Wire.begin();

  if (! setvolume(thevol)) {
    Serial.println("Failed to set volume, MAX9744 not found!");
    while (1);
  }
}


// Setting the volume is very simple! Just write the 6-bit
// volume to the i2c bus. That's it!
boolean setvolume(int8_t v) {
  // cant be higher than 63 or lower than 0
  if (v > 63) v = 63;
  if (v < 0) v = 0;

  Serial.print("Setting volume to ");
  Serial.println(v);
  Wire.beginTransmission(MAX9744_I2CADDR);
  Wire.write(v);
  if (Wire.endTransmission() == 0)
    return true;
  else
    return false;
}

// Read in + and - characters to set the volume.
void loop() {
  if (! Serial.available()) return;

  // read a character from serial console
  char c = Serial.read();

  // increase
  if (c == '+') {
    thevol++;
  }
  // decrease
  else if (c == '-') {
    thevol--;
  }
  // ignore anything else
  else
    return;
```

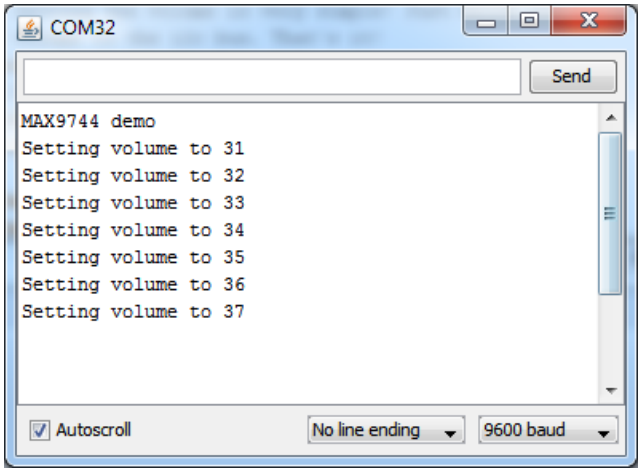
```
    return;

  if (thevol > 63) thevol = 63;
  if (thevol < 0) thevol = 0;

  setvolume(thevol);
}
```

Run the sketch and open up your Arduino Serial console. If there's a problem you'll get a failure to set the volume initially



If it's OK, you can send multiple +'s and -'s to increase or decrease the volume. The volume can go down to 0 or up to 63 (which is 29dB of gain)





Changing the I2C address

if you need to change the address from the 0x4B default to something else, you can close the AD1 or AD2 jumpers **but not both** to change the address to 0x4A (AD1 closed or tied to ground) or 0x49 (AD2 closed or tied to ground)

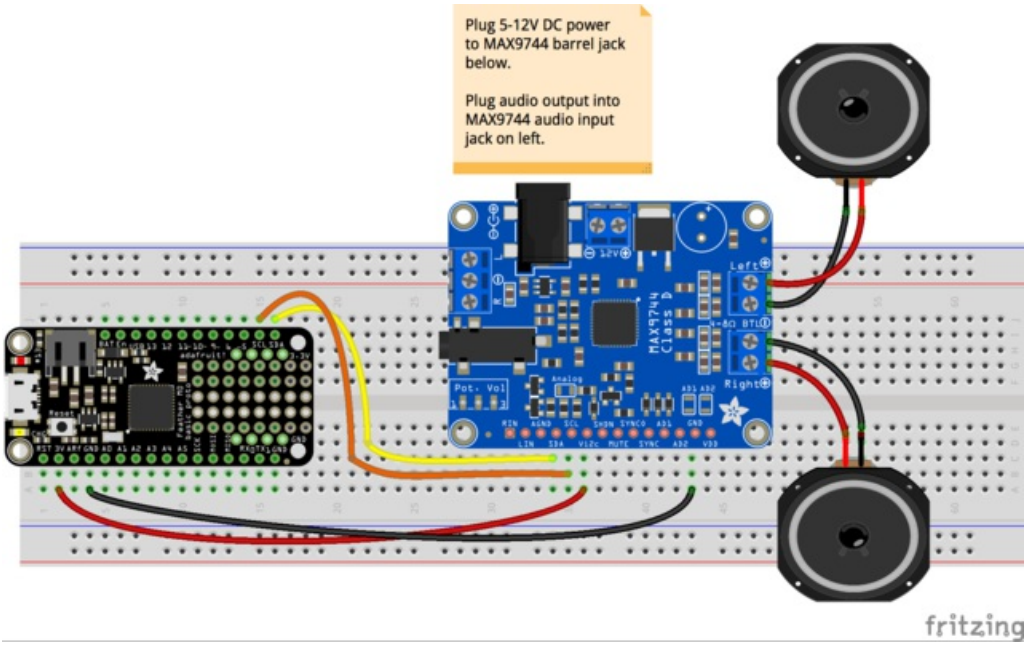
Connecting to a Raspberry Pi or BeagleBone Black

If you'd like to control the MAX9744 from a Raspberry Pi or BeagleBone Black then check out this Adafruit MAX9744 Python library. The readme for the library has instructions on connecting to the board, installing the library, and demonstrating the usage with an example. Using this library you can set the volume and increase or decrease it using Python and an I2C connection from your board.

CircuitPython Control

It's easy to use the MAX9744 amplifier with CircuitPython and the Adafruit CircuitPython MAX9744 module. This module allows you to easily write Python code that controls the volume of the amplifier over its I2C connection.

First wire up a MAX9744 to your board exactly as shown on the previous pages for Arduino using an I2C connection. Here's an example of wiring a Feather M0 to the sensor with I2C:



- **Board 3V** to **sensor Vi2c**
- **Board GND** to **sensor GND**
- **Board SCL** to **sensor SCL**
- **Board SDA** to **sensor SDA**

In addition just like the basic test page mentions be sure to also wire up a power supply, speakers, and audio input to the amplifier.

Next you'll need to install the Adafruit CircuitPython MAX9744 library on your CircuitPython board.

First make sure you are running the latest version of Adafruit CircuitPython for your board.

Next you'll need to install the necessary libraries to use the hardware--carefully follow the steps to find and install these libraries from Adafruit's CircuitPython library bundle. Our introduction guide has a great page on how to install the library bundle for both express and non-express boards.

Remember for non-express boards like the, you'll need to manually install the necessary libraries from the bundle:

- **adafruit_max9744.mpy**

You can also download the **adafruit_max9744.mpy** from its releases page on Github.

Before continuing make sure your board's lib folder or root filesystem has the **adafruit_max9744.mpy** file copied over.

Next connect to the board's serial REPL so you are at the CircuitPython >>> prompt.

Usage

To demonstrate the usage of the amplifier we'll initialize it and control the volume from the board's Python REPL. Run the following code to import the necessary modules and initialize the I2C connection with the amplifier:

```
import board
import busio
import adafruit_max9744
i2c = busio.I2C(board.SCL, board.SDA)
amp = adafruit_max9744.MAX9744(i2c)
```

Remember if you're using a board that doesn't support hardware I2C (like the ESP8266) you need to use the **bitbangio** module instead:

```
import board
import bitbangio
import adafruit_max9744
i2c = bitbangio.I2C(board.SCL, board.SDA)
amp = adafruit_max9744.MAX9744(i2c)
```

Make sure some audio is playing through the amplifier, then you can control the volume using a few commands.

The first option is to set the **volume** property to an explicit value. This can be any number from 0 to 63 where 0 is off/muted and 63 is maximum intensity (**be careful, this amplifier can produce 20 watts of output which might damage small speakers!**)

For example to set a moderate half-way to max volume:

```
amp.volume = 31
```

Or to mute/turn off the output:

```
amp.volume = 0
```

In addition you can call the **volume_up** and **volume_down** functions to tell the amp to move up or down a single volume level. This might be handy if your project only has an up/down volume control.

For example to move up 2 levels and then back down 1 level:

```
amp.volume_up()
amp.volume_up()
amp.volume_down()
```

Again be careful to not increase the volume to such a high level that it damages your speakers!

That's all there is to using the MAX9744 amplifier with CircuitPython!

Below is a complete example of setting the volume of the amplifier. Save this as **main.py** on the board and it will set the volume to a moderate/half-way level. Be sure to modify it to use the **bitbangio** module if necessary!

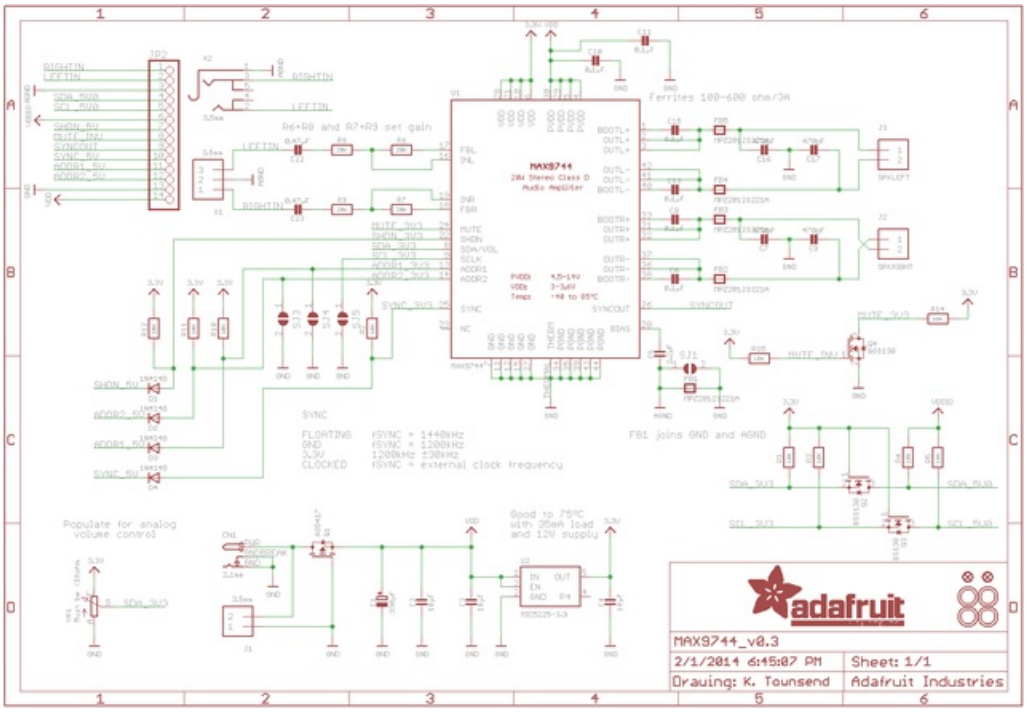
```
# Simple demo of the MAX9744 20W class D amplifier I2C control.
# This show how to set the volume of the amplifier.
# Author: Tony DiCola
import board
import busio

import adafruit_max9744


# Initialize I2C bus.
i2c = busio.I2C(board.SCL, board.SDA)

# Initialize amplifier.
amp = adafruit_max9744.MAX9744(i2c)
# Optionally you can specify a different addres if you override the AD1, AD2
# pins to change the address.
#amp = adafruit_max9744.MAX9744(i2c, address=0x49)

# Setting the volume is as easy as writing to the volume property (note
# you cannot read the property so keep track of volume in your own code if
# you need it).
amp.volume = 31  # Volume is a value from 0 to 63 where 0 is muted/off and
                 # 63 is maximum volume.

# In addition you can call a function to instruct the amp to move up or down
# a single volume level.  This is handy if you just have up/down buttons in
# your project for volume:
amp.volume_up()    # Increase volume by one level.

amp.volume_down()  # Decrease volume by one level.
```

Downloads

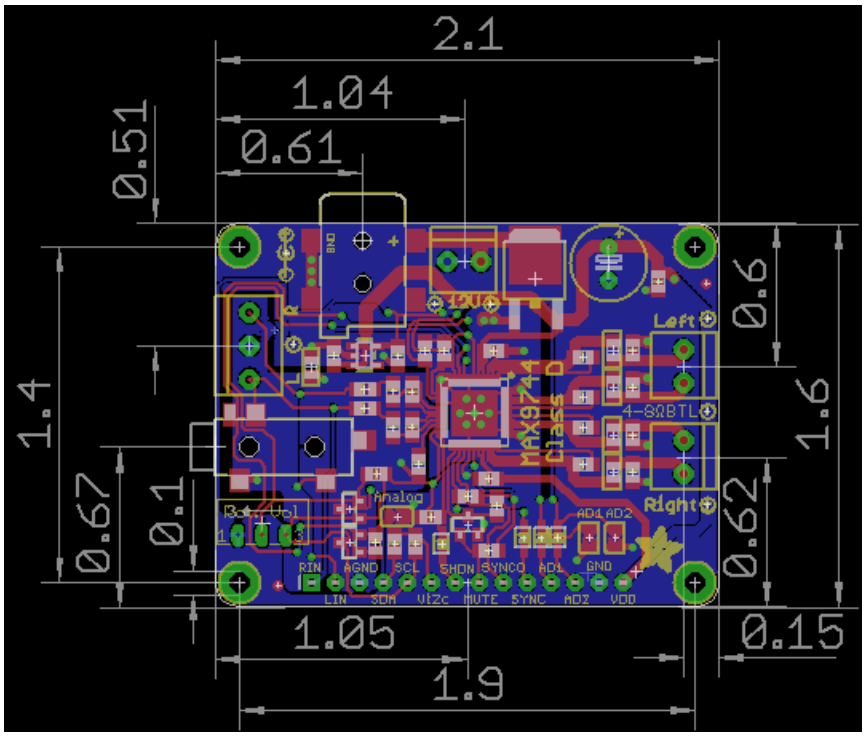
Datasheets & Files

- Datasheet for the MAX9744
- Fritzing object in Adafruit Fritzing library
- EagleCAD PCB files on GitHub

Schematic



Layout Dimensions


2.1
1.04
0.61
0.51
1.4
0.67
0.1
0.6
1.6
0.62
0.15
1.05
1.9
Left
4-8ΩBTL
Right
MAX9744
Class D
Analog
AD1 AD2
RIN AGND SCL SHDN SYNCO ADR GND
LIN SDA VI2C MUTE SYNC ADI VDD