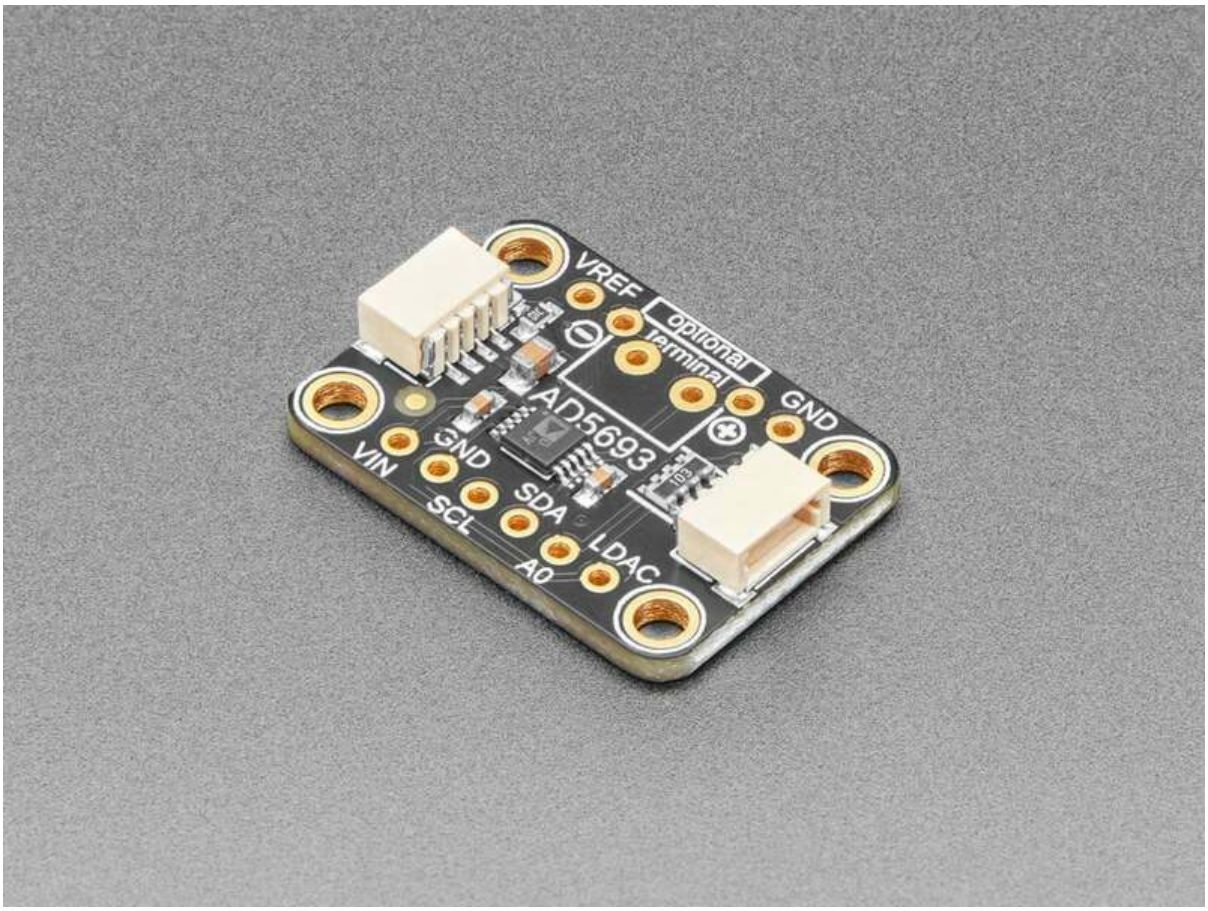
Adafruit AD5693R 16-Bit DAC Breakout Board

Created by Liz Clark

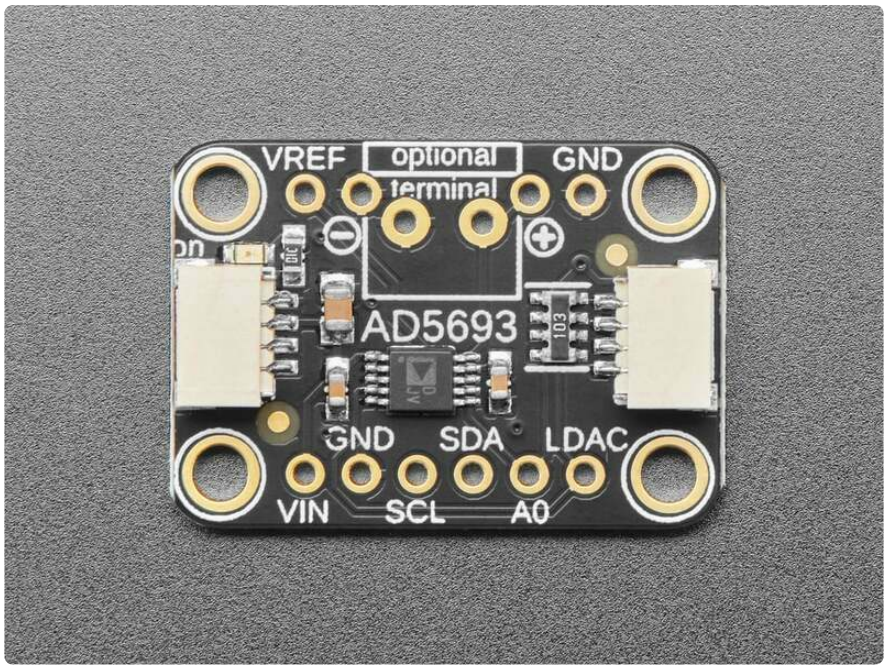


https://learn.adafruit.com/adafruit-ad5693r-16-bit-dac-breakout-board

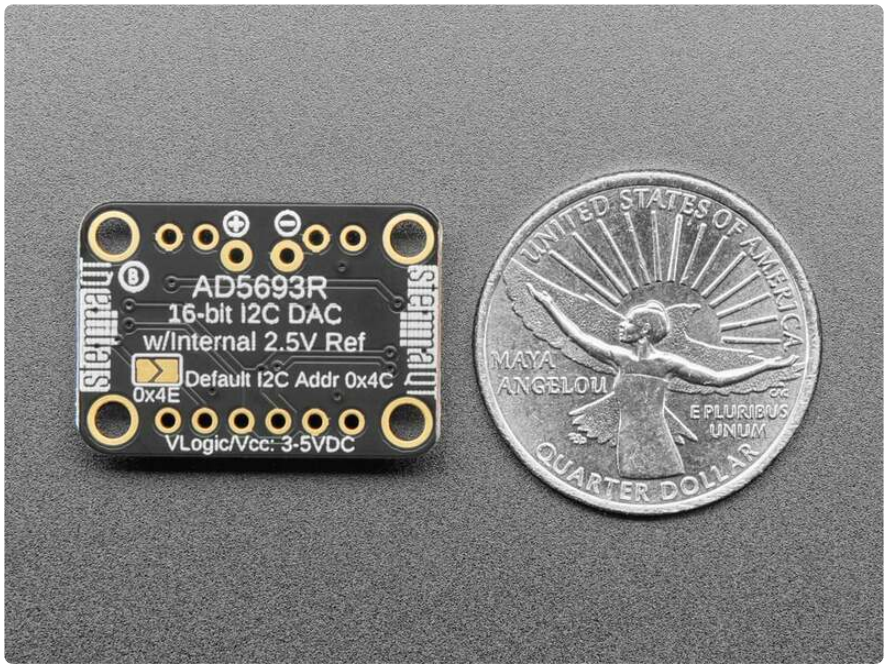
Last updated on 2024-06-03 03:55:30 PM EDT

Table of Contents

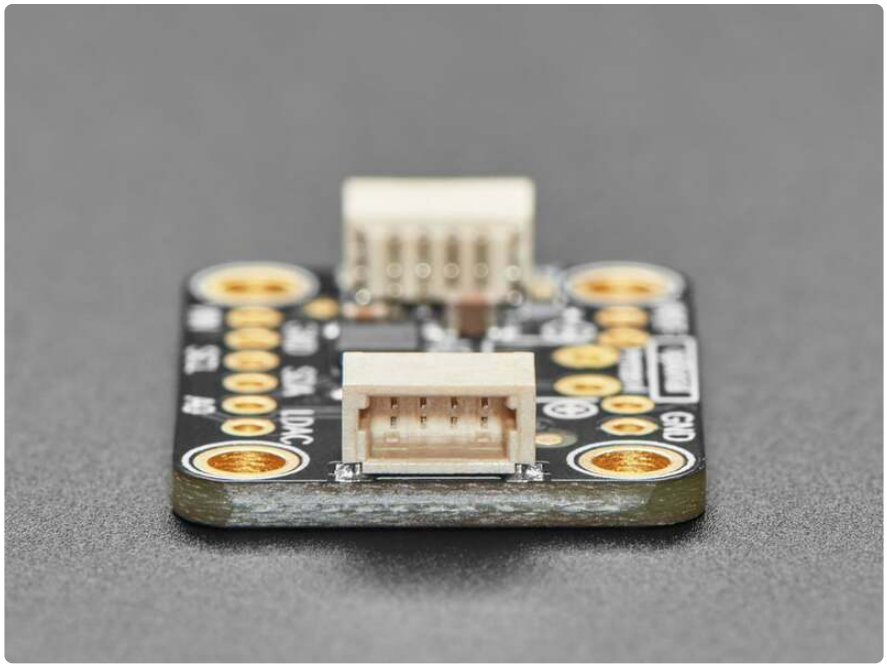
Overview



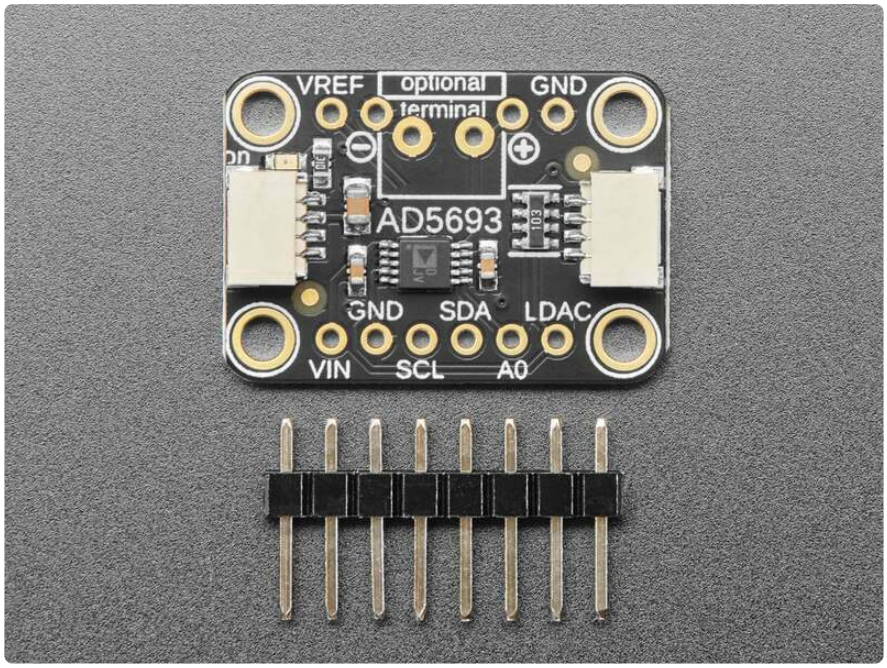
Which is better, less bits or more? MORE of course! So why settle for a 12-bit DAC like the MCP4725 (http://adafru.it/935) when you can go for the 16-bits of the AD5693? OK, well there may be some reason to go with 12-bits, say if you don't need high resolution output and want to go with the more affordable DAC. But for those who like the finer things in life, the **Adafruit AD5693R Breakout Board is a 16-Bit DAC with I2C Interface** and temperature compensated 2.5V internal reference for a compact high-precision output.



We break out the ADDR/A0 pin so you can connect two of these DACs on one I2C bus, just tie the A0 pin high (or close the jumper on the back) to keep it from conflicting. Also included is a 6-pin header, for use in a breadboard. Works with both 3.3V or 5V logic, and you can have the output max out at 2.5V or 5V (2xVref). If you're powering from 3.3V, you will be able to set the output range to 2.5V or Vin.



We have an easy-to-use Arduino library and tutorial with a sine-wave output example (https://adafru.it/194a) that can be used with just about any microcontroller or microcomputer with I2C host.

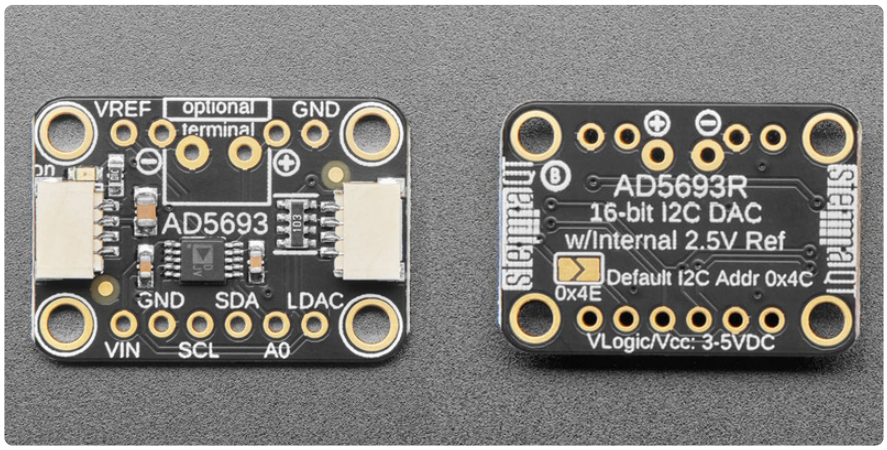


Comes with a bit of 0.1" standard header in case you want to use it with a breadboard or perfboard. Four mounting holes for easy attachment. There's an optional 3.5mm

terminal block spot on the PCB - we don't include a 3.5mm terminal block but they're both common and stocked in the shop (http://adafru.it/724) - that you can solder in place if you like.

To get you going fast, we spun up a custom-made PCB in the **STEMMA QT** form factor (https://adafru.it/LBQ), making it easy to interface with. The STEMMA QT connectors (https://adafru.it/JqB) on either side are compatible with the SparkFun Qwiic (https://adafru.it/Fpw) I2C connectors. This allows you to make solderless connections between your development board and the AD5693R or to chain it with a wide range of other sensors and accessories using a **compatible cable** (https://adafru.it/JnB).

Pinouts



The default I2C address is **0x4C**.

Power Pins

- **VIN** - this is the power pin. Since the DAC chip uses 3-5 VDC to power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V micro like Arduino, use 5V.
- **VREF** - this is the voltage reference pin.
- **GND** - common ground for power and logic.

I2C Logic Pins

- **SCL** - I2C clock pin, connect to your microcontroller's I2C clock line. This pin can use 3-5V logic, and there's a **10K pullup** on this pin.
- **SDA** - I2C data pin, connect to your microcontroller's I2C data line. This pin can use 3-5V logic, and there's a **10K pullup** on this pin.

- STEMMA QT **(https://adafru.it/Ft4)** - These connectors allow you to connectors to dev boards with **STEMMA QT** connectors or to other things with various associated accessories (https://adafru.it/Ft6)

DAC Output

- + - The analog voltage output from the DAC.

Address Pin and Jumper

- **A0** - The address pin for setting the I2C address. You can chain up to two of these boards together on one I2C bus. Leave this pin **low** for default I2C address **0x4C** or tie it **high** for I2C address **0x4E**.
- **0x4E Jumper** - On the back of the board is the address jumper, labeled **0x4E**. You can leave this jumper open (low) to keep the board on the default I2C address 0x4C. Solder the jumper closed (high) to change the I2C address to 0x4E.

ADDR	A0
0x4C	L
0x4E	H

LDAC Pin

- **LDAC** - The load DAC pin. Transfers the content of the input register to the DAC register. The pin is tied to ground, allowing the DAC to update when new data is written to the input register.

Power LED

- **Power LED -** In the upper left corner, above the STEMMA connector, on the front of the board, is the power LED, labeled **on**. It is the green LED.

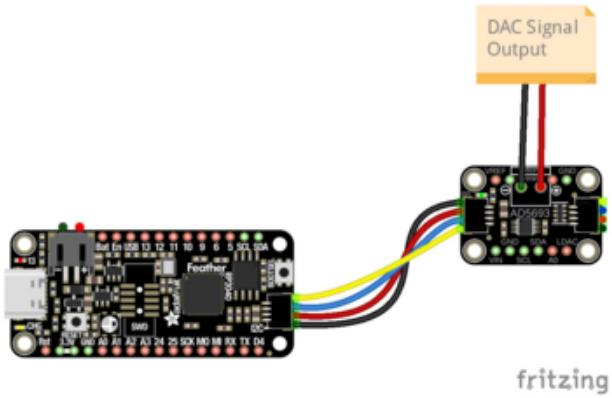
CircuitPython and Python

It's easy to use the **AD5693R** with Python or CircuitPython, and the Adafruit_CircuitPython_AD569x (https://adafru.it/194b) module. This module allows you to easily write Python code to control the DAC.

You can use this driver with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library (https://adafru.it/BSN).

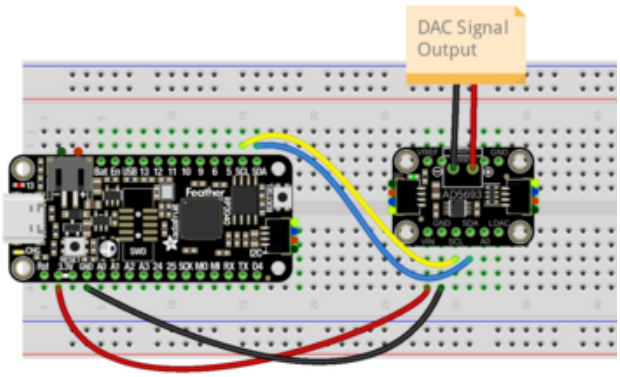
CircuitPython Microcontroller Wiring

First wire up the breakout to your board exactly as follows. The following is the breakout wired to a Feather RP2040 using the STEMMA connector:



Board STEMMA 3V to **breakout VIN (red wire)**
Board STEMMA GND to **breakout GND (black wire)**
Board STEMMA SCL to **breakout SCL (yellow wire)**
Board STEMMA SDA to **breakout SDA (blue wire)**
The signal will be output from the DAC **Vout (+)** pin.

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

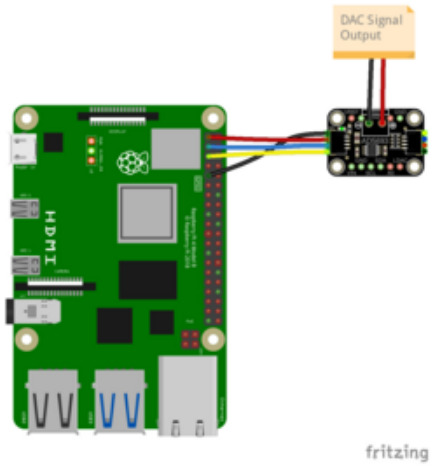


Board 3V to **breakout VIN (red wire)**
Board GND to **breakout GND (black wire)**
Board SCL to **breakout SCL (yellow wire)**
Board SDA to **breakout SDA (blue wire)**
The signal will be output from the DAC **Vout (+)** pin.

Python Computer Wiring

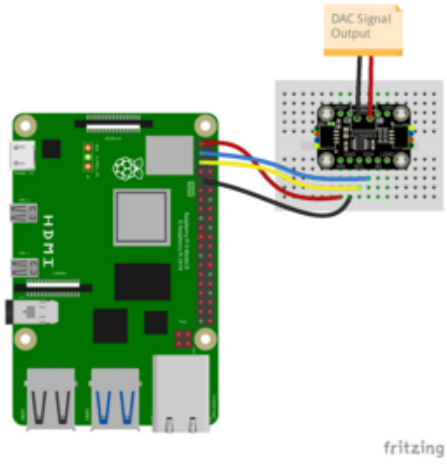
Since there are dozens of Linux computers/boards you can use, we will show wiring for Raspberry Pi. For other platforms, [please visit the guide for CircuitPython on Linux to see whether your platform is supported](https://adafru.it/BSN).

Here's the Raspberry Pi wired with I2C using the STEMMA connector:



Pi 3V to **breakout VIN (red wire)**
Pi GND to **breakout GND (black wire)**
Pi SCL to **breakout SCL (yellow wire)**
Pi SDA to **breakout SDA (blue wire)**
The signal will be output from the DAC **Vout (+)** pin.

Here's the Raspberry Pi wired with I2C using a solderless breadboard:



Pi 3V to **breakout VIN (red wire)**
Pi GND to **breakout GND (black wire)**
Pi SCL to **breakout SCL (yellow wire)**
Pi SDA to **breakout SDA (blue wire)**
The signal will be output from the DAC **Vout (+)** pin.

Python Installation of AD569x Library

You'll need to install the **Adafruit_Blinka** library that provides the CircuitPython support in Python. This may also require enabling I2C on your platform and verifying you are running Python 3. [Since each platform is a little different, and Linux changes often, please visit the CircuitPython on Linux guide to get your computer ready](https://adafru.it/BSN)!

Once that's done, from your command line run the following command:

- `pip3 install adafruit-circuitpython-ad569x`

If your default Python is version 3 you may need to run 'pip' instead. Just make sure you aren't trying to use CircuitPython on Python 2.x, it isn't supported!

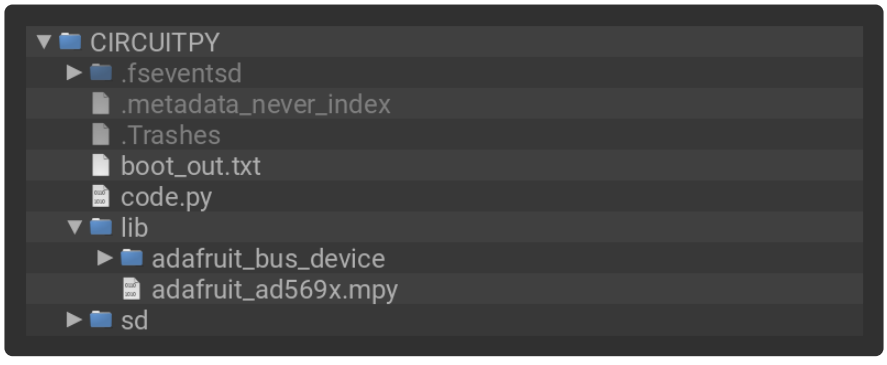
CircuitPython Usage

To use with CircuitPython, you need to first install the **Adafruit_CircuitPython_AD569x** library, and its dependencies, into the **lib** folder on your **CIRCUITPY** drive. Then you need to update **code.py** with the example script.

Thankfully, we can do this in one go. In the example below, click the **Download Project Bundle** button below to download the necessary libraries and the **code.py** file in a zip file. Extract the contents of the zip file, and copy the **entire lib folder** and the **code.py** file to your **CIRCUITPY** drive.

Your **CIRCUITPY/lib** folder should contain the following folders and file:

- **adafruit_bus_device/**
- **adafruit_register/**
- **adafruit_ad569x.mpy**



Python Usage

Once you have the library `pip3` installed on your computer, copy or download the following example to your computer, and run the following, replacing **code.py** with whatever you named the file:

```
python3 code.py
```

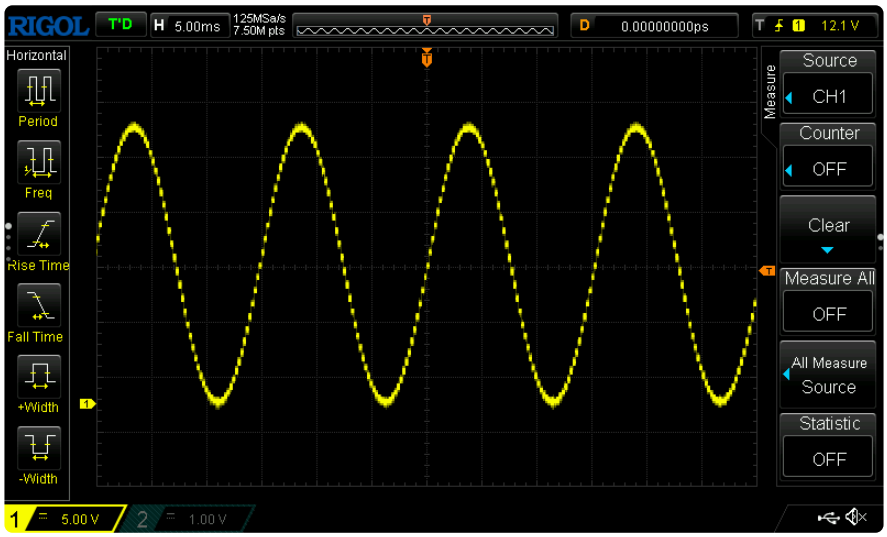
Example Code

If running CircuitPython: Once everything is saved to the **CIRCUITPY** drive, connect to the serial console (https://adafru.it/Bec) to see the data printed out!

If running Python: The console output will appear wherever you are running Python.

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

"""Simple demo of writing a sine wave to the AD569x DAC."""

import math
import board
import busio
import adafruit_ad569x

i2c = busio.I2C(board.SCL, board.SDA, frequency=400_000)

# Initialize AD569x
dac = adafruit_ad569x.Adafruit_AD569x(i2c)

# length of the sine wave
LENGTH = 100
# sine wave values written to the DAC
value = [
    int(math.sin(math.pi * 2 * i / LENGTH) * ((2**15) - 1) + 2**15)
    for i in range(LENGTH)
]

while True:
    for v in value:
        dac.value = v
```

The AD5693 is initialized over I2C. Then in the loop, a sine wave is output to the DAC. If you connect the DAC output to a scope or other analog visualizer, you will see the sine wave displayed.



Python Docs

Python Docs (https://adafru.it/194c)

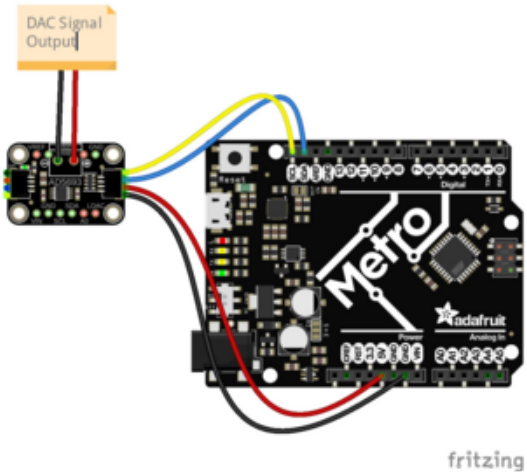
Arduino

Using the AD5693R breakout with Arduino involves wiring up the breakout to your Arduino-compatible microcontroller, installing the Adafruit_AD569x (https://adafru.it/194f) library, and running the provided example code.

Wiring

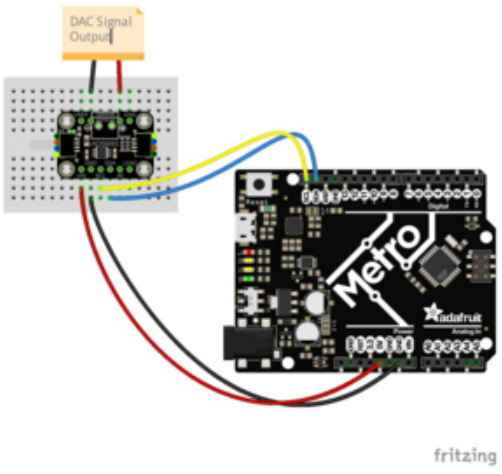
Wire as shown for a **5V** board like an Uno. If you are using a **3V** board, like an Adafruit Feather, wire the board's 3V pin to the breakout VIN.

Here is an Adafruit Metro wired up to the breakout using the STEMMA QT connector:



Board 5V to **breakout VIN (red wire)**
Board GND to **breakout GND (black wire)**
Board SCL to **breakout SCL (yellow wire)**
Board SDA to **breakout SDA (blue wire)**
The signal will be output from the DAC **Vout** (+) pin.

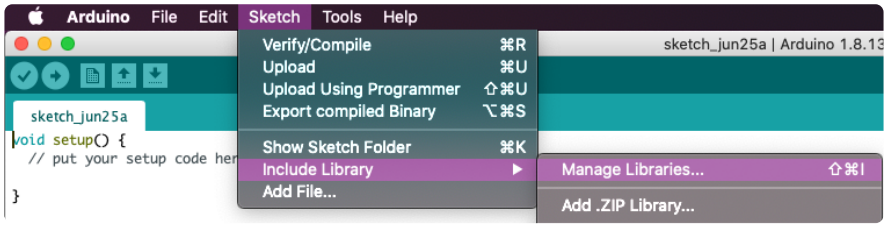
Here is an Adafruit Metro wired up using a solderless breadboard:



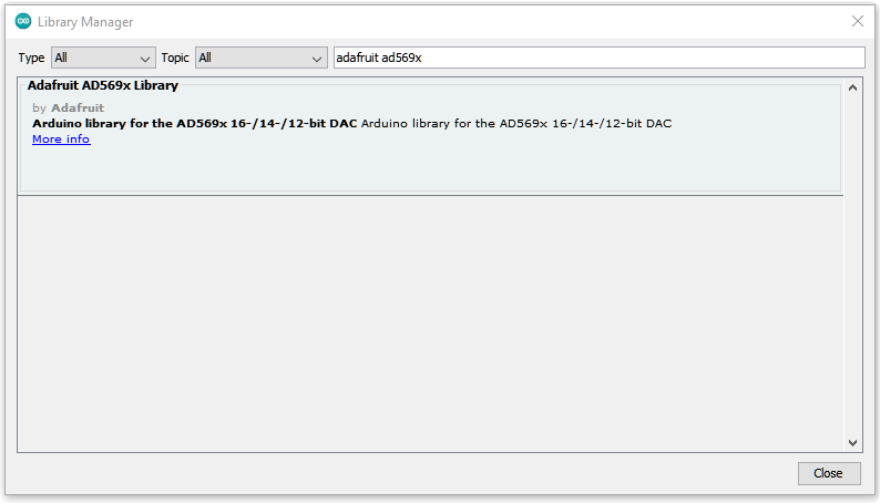
Board 5V to **breakout VIN (red wire)**
Board GND to **breakout GND (black wire)**
Board SCL to **breakout SCL (yellow wire)**
Board SDA to **breakout SDA (blue wire)**
The signal will be output from the DAC **Vout** (+) pin.

Library Installation

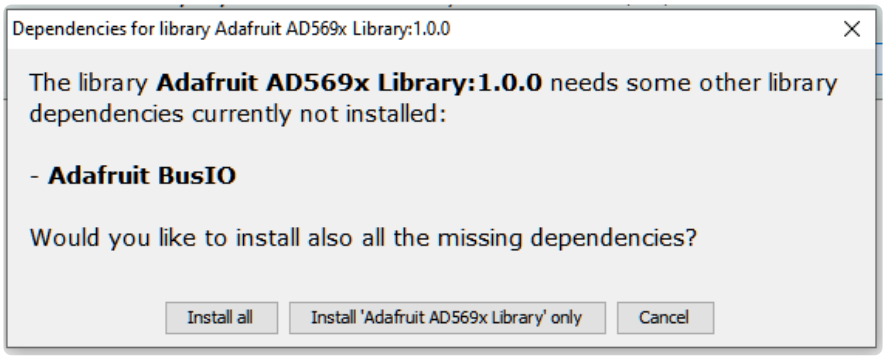
You can install the **Adafruit_AD569x** library for Arduino using the Library Manager in the Arduino IDE.



Click the **Manage Libraries ...** menu item, search for **Adafruit_AD569x**, and select the **Adafruit AD569x** library:



If asked about dependencies, click "Install all".



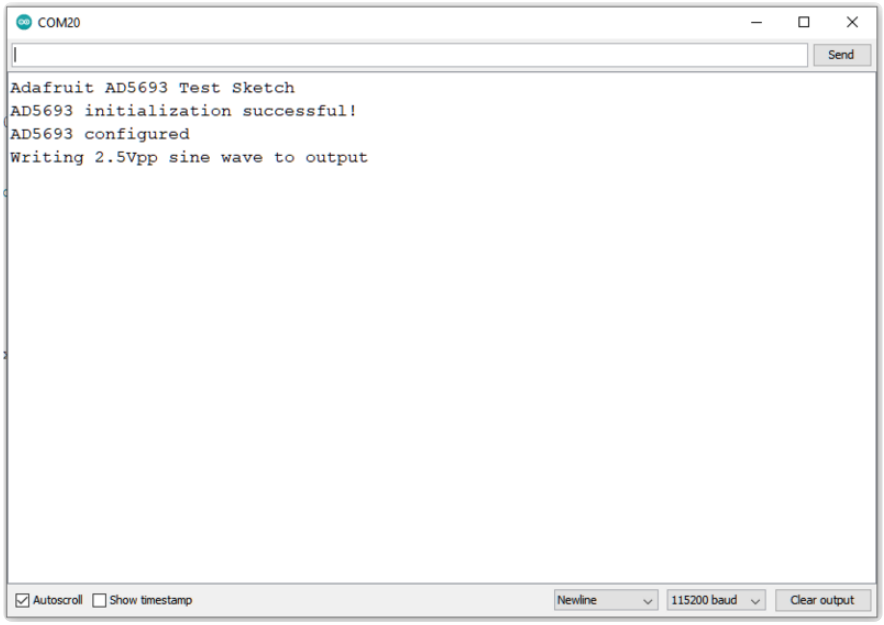
If the "Dependencies" window does not come up, then you already have the dependencies installed.

If the dependencies are already installed, you must make sure you update them through the Arduino Library Manager before loading the example!

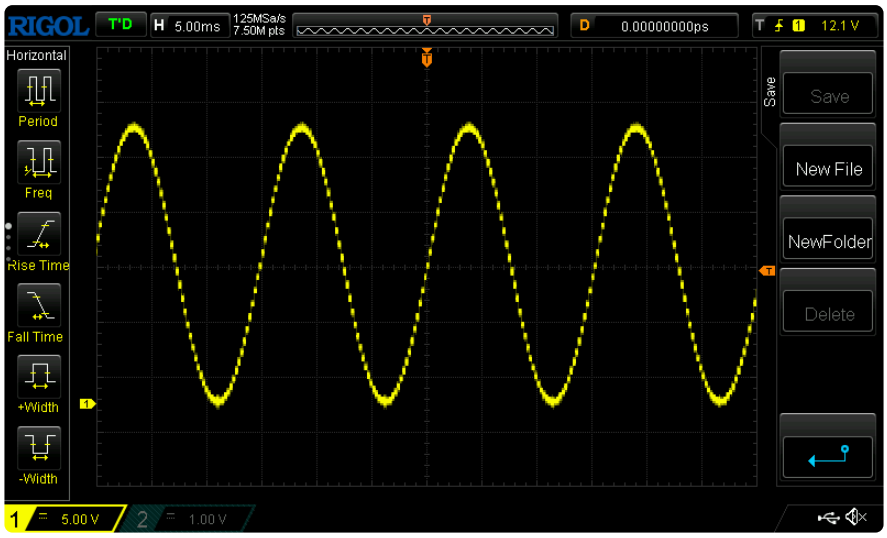
Example Code

```
#include "Adafruit_AD569x.h"
#include <math.h> // For sine function

Adafruit_AD569x ad5693; // Create an object of the AD5693 library

void setup() {
  Serial.begin(115200);
  while (!Serial) delay(10); // Wait for serial port to start
  Serial.println("Adafruit AD5693 Test Sketch");

  // Initialize the AD5693 chip
  if (ad5693.begin(0x4C, &Wire)) { // If A0 jumper is set high, use 0x4E
    Serial.println("AD5693 initialization successful!");
  } else {
    Serial.println("Failed to initialize AD5693. Please check your connections.");
    while (1) delay(10); // Halt
  }

  // Reset the DAC
  ad5693.reset();

  // Configure the DAC for normal mode, internal reference, and no 2x gain
  if (ad5693.setMode(NORMAL_MODE, true, false)) {
    Serial.println("AD5693 configured");
  } else {
    Serial.println("Failed to configure AD5693.");
    while (1) delay(10); // Halt
  }

  // You probably will want to set the I2C clock rate to faster
  // than the default 100KHz, try 400K or 800K or even 1M!
  Wire.setClock(800000);

  Serial.println("Writing 2.5Vpp sine wave to output");
}

void loop() {
  // Generate a sine wave and write it to the DAC
  for (float angle = 0; angle <= 2 * PI; angle += 0.1) {
    uint16_t value = (uint16_t)((sin(angle) + 1) * 32767.5); // Convert sine value
to uint16_t
    if (!ad5693.writeUpdateDAC(value)) {
      Serial.println("Failed to update DAC.");
    }
  }
}
```

Upload the sketch to your board and open up the Serial Monitor (**Tools -> Serial Monitor**) at 115200 baud. You'll see the AD5693R recognized over I2C.



If you connect the DAC output to a scope or other analog visualizer, you will see a sine wave displayed.



Arduino Docs

Arduino Docs (https://adafru.it/194A)

Downloads

Files

- AD5693R Datasheet (https://adafru.it/194B)
- EagleCAD PCB Files on GitHub (https://adafru.it/194C)

- Fritzing object in the Adafruit Fritzing Library (https://adafru.it/194F)

Schematic and Fab Print