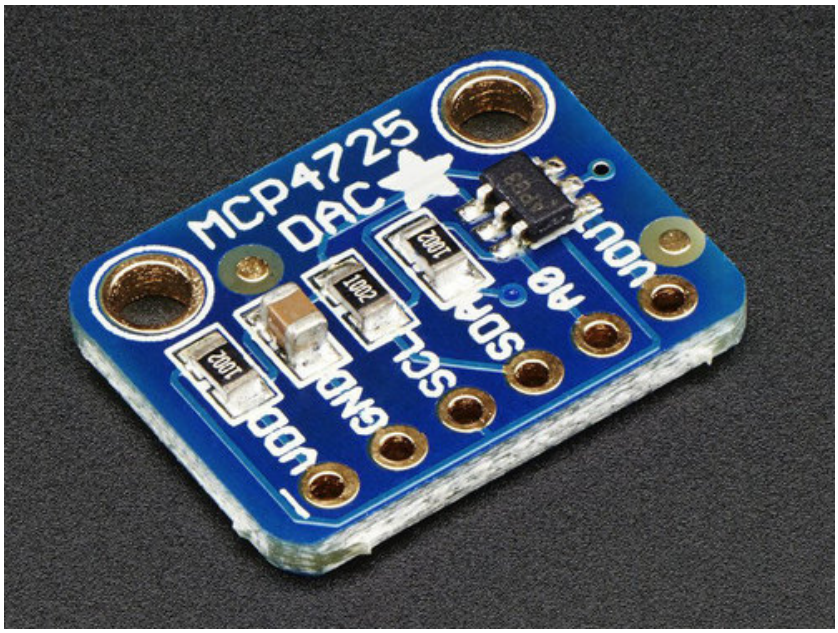
MCP4725 12-Bit DAC Tutorial

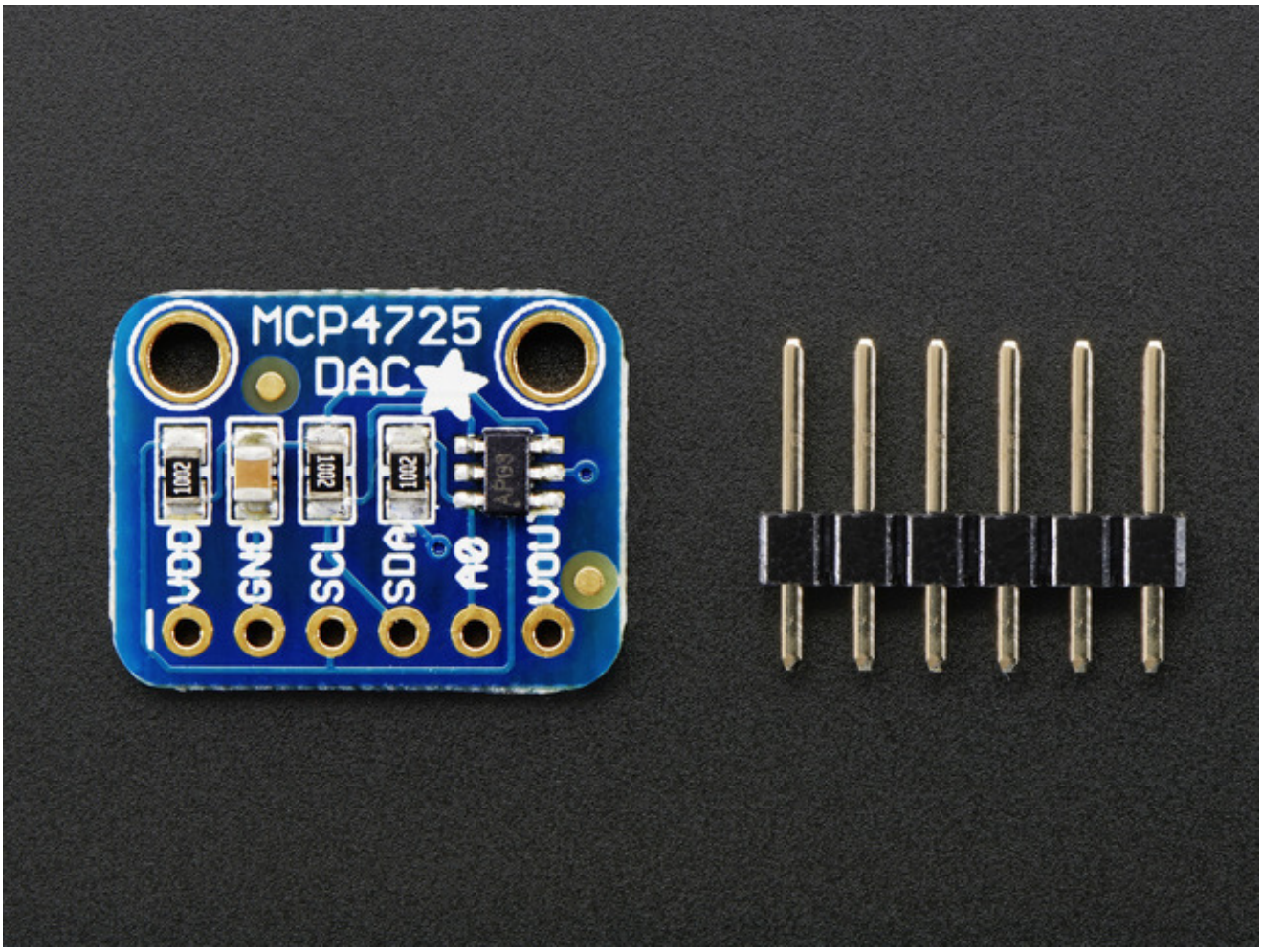
Created by lady ada



Last updated on 2016-10-07 04:47:03 PM UTC

Guide Contents

Overview



Your microcontroller probably has an ADC (analog -> digital converter) but does it have a DAC (digital -> analog converter)??? Now it can! This breakout board features the easy-to-use MCP4725 12-bit DAC. Control it via I2C and send it the value you want it to output, and the VOUT pin will have it. Great for audio / analog projects, such as when you can't use PWM but need a sine wave or adjustable bias point.

We break out the ADDR pin so you can connect two of these DACs on one I2C bus, just tie the ADDR pin of one high 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.

Some nice extras with this chip: for chips that have 3.4Mbps Fast Mode I2C (Arduino's don't) you can update the Vout at ~200 KHz. There's an EEPROM so if you write the output voltage, you can 'store it' so if the device is power cycled it will restore that voltage. The output voltage is rail-to-rail and proportional to the power pin so if you run it from 3.3V, the

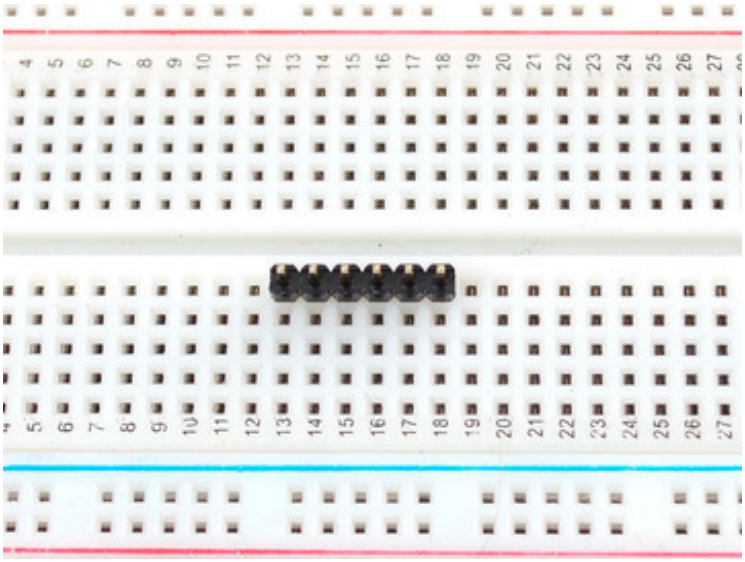
output range is 0-3.3V. If you run it from 5V the output range is 0-5V.

Available from the Adafruit shop! (http://adafru.it/935)

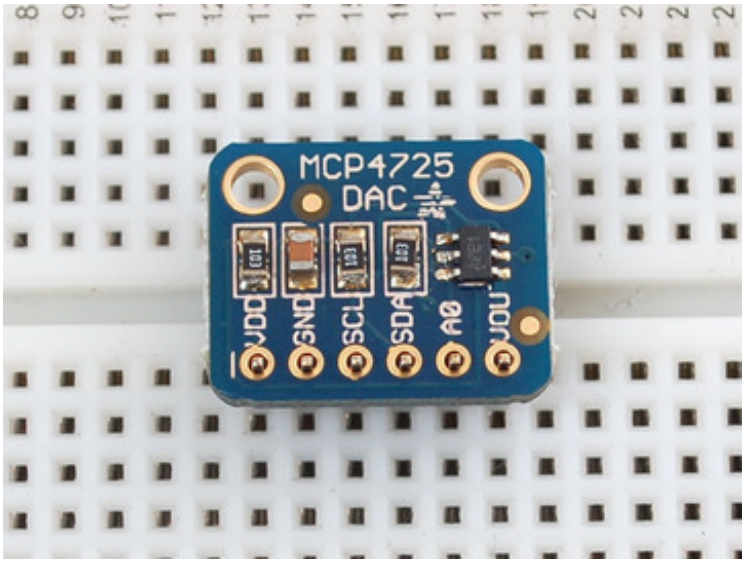
Wiring

Wiring up the MCP4725 breakout PCB is super easy. To start, we'll attach the breakout headers so we can plug it into a breadboard.

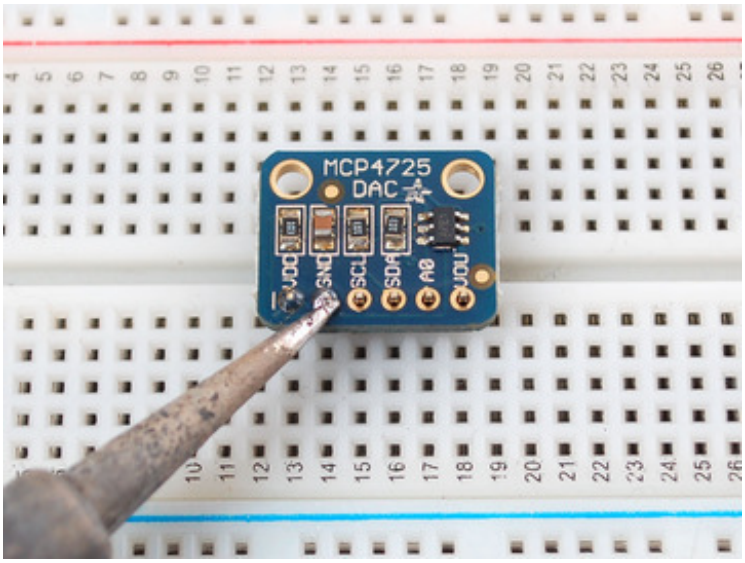
Break off a strip of 6-pins of 0.1" male header and stick the LONG pins down into a breadboard

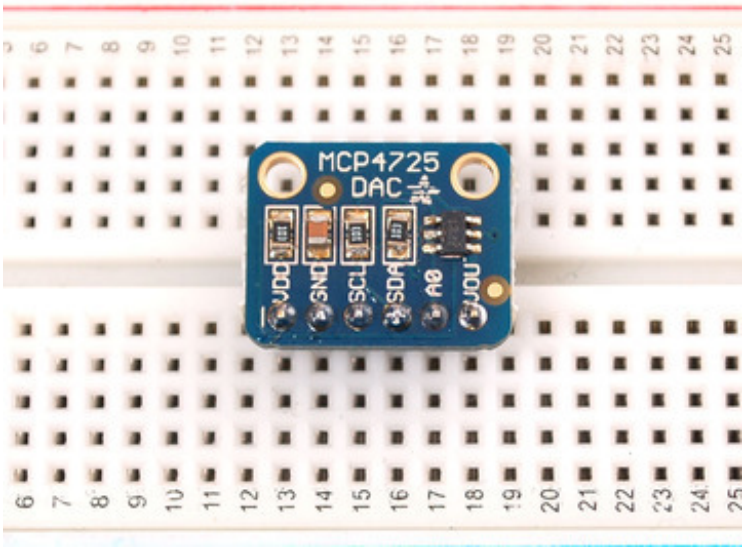
- 

Break off a strip of 6-pins of 0.1" male header and stick the LONG pins down into a breadboard

- 

Place the breakout board on top so the short ends of the header stick up through the pads

- 

Solder each pin using a soldering iron and solder, to make solid connection on each pin.
- 

This part is not optional! You cannot 'press fit' the header on, it must be attached permanently

Now that the header is attached, we can wire it up. We'll demonstrate using an Arduino.

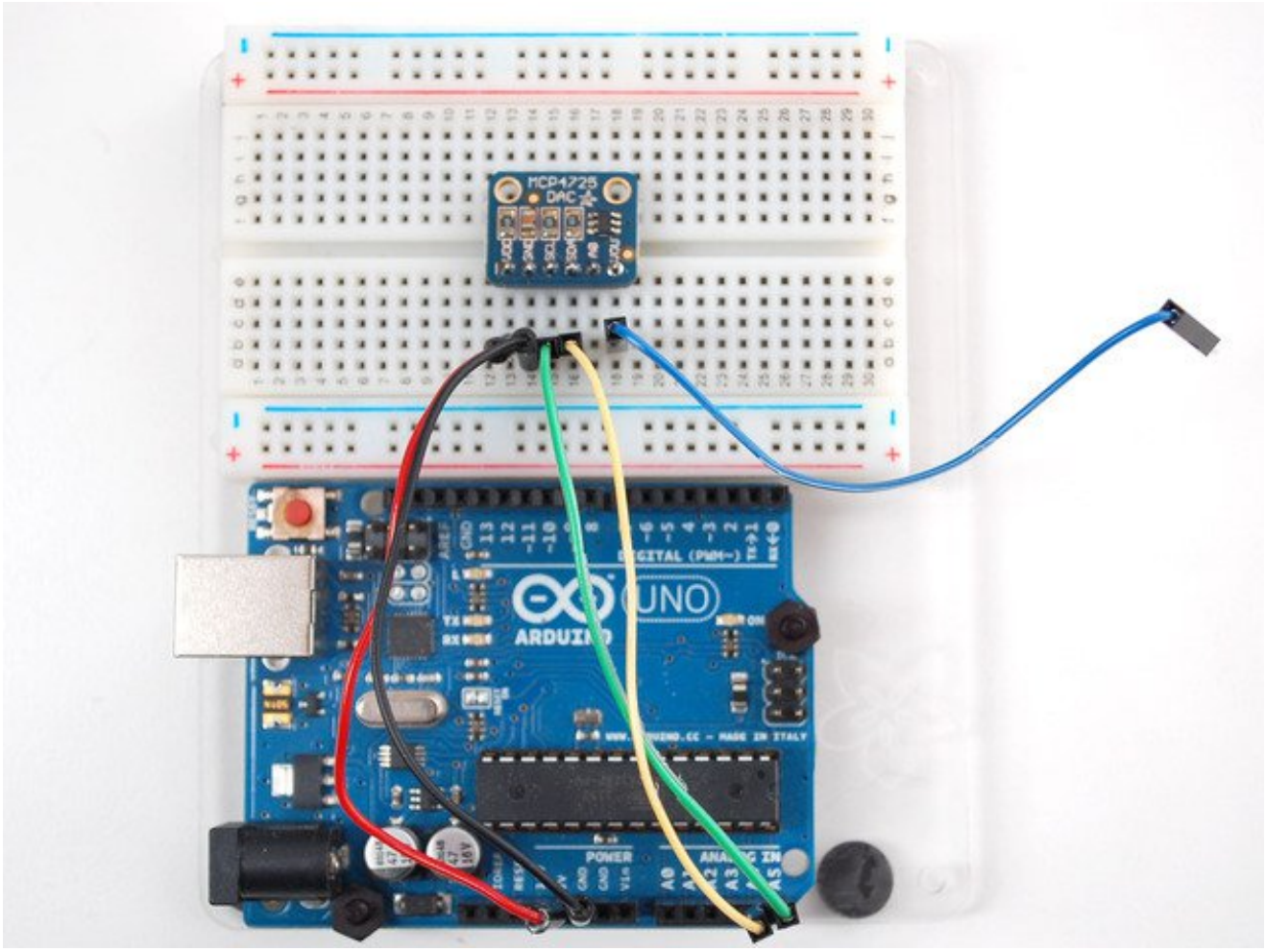
FIrst, connect **VDD** (power) to a 3-5V power supply, and **GND** to ground.

The DAC uses I2C, a two-pin interface that can have up to 127 unique sensors attached (each must have a different **ADDRESS**).

- **SDA** to I2C Data (on the Uno, this is **A4** on the Mega it is **20** and on the Leonardo digital **2**)
- **SCL** to I2C Clock(on the Uno, this is **A5** on the Mega it is **21** and on the Leonardo digital **3**)

There's two other pins remaining.

- **A0** allow you to change the I2C address. By default (nothing attached to A0) the address is hex **0x62**. If A0 is connected to **VDD** the address is **0x63**. This lets you have two DAC boards connected to the same SDA/SCL I2C bus pins.
- **VOUT** is the voltage out from the DAC! The voltage will range from 0V (when the DAC value is 0) to VDD (when the DAC 'value' is the max 12-bit number: 0xFFF)



Using with Arduino

Next up, download the Adafruit MCP4725 library. This library does all of the interfacing, so you can just "set and forget" the DAC output. It also has some examples to get you started

The library is available on GitHub (http://adafru.it/aPz). You can download it by clicking the button below.

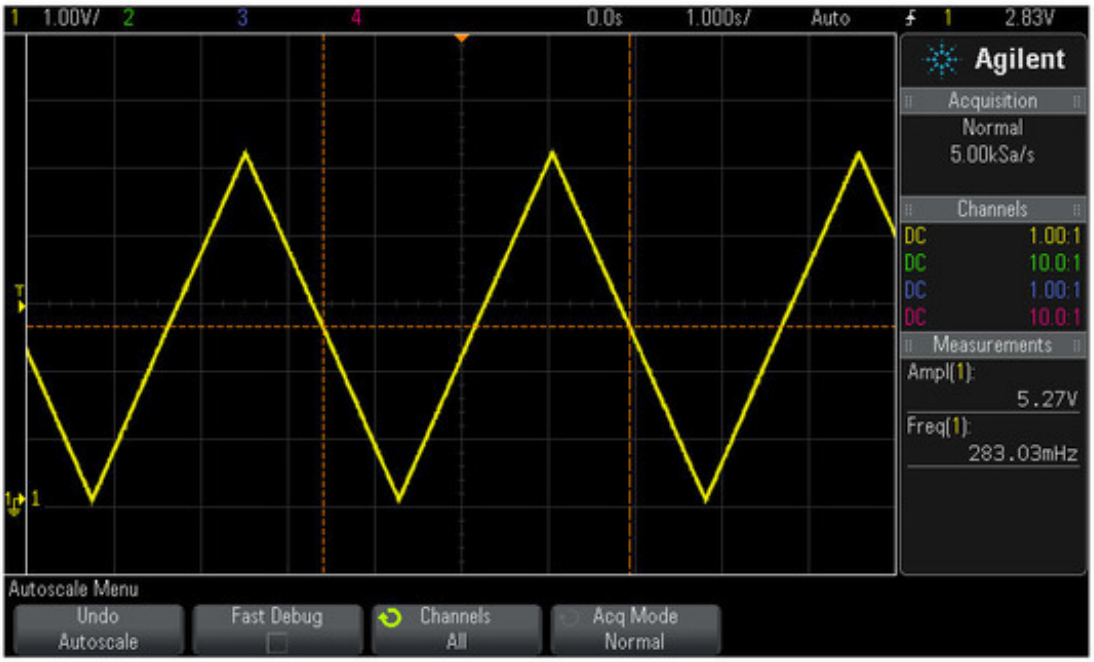
Download Adafruit_MCP4725 Library
http://adafru.it/cDA

Rename the uncompressed folder **Adafruit_MCP4725.** Check that the **Adafruit_MCP4725** folder contains **Adafruit_MCP4725.cpp** and **Adafruit_MCP4725.h**

Place the **Adafruit_MCP4725** library folder your ***sketchbookfolder*/libraries/** folder. You may need to create the **libraries** subfolder if its your first library. You can figure out your ***sketchbookfolder*** by opening up the Preferences tab in the Arduino IDE.

Restart the IDE.

Open up the **File→Examples→Adafruit_MCP4725→trianglewave** sketch and upload it to the Arduino. Then connect your oscilloscope (or an LED + resistor if you don't have access to an oscilloscope)



We also have a sine wave version showing how to use a lookup table to create a more complex waveform.

Using the library

The library is very simple, so you can adapt it very quickly.

First, be sure to call **begin(addr)** where **addr** is the i2c address (default is 0x62, if A0 is connected to VCC its 0x63). Then call **setVoltage(value, storeflag)** to set the DAC output. **value** should range from 0 to 0x0FFF. **storeflag** indicates to the DAC whether it should store the value in EEPROM so that next time it starts, it'll have that same value output. You shouldn't set the flag to true unless you require it as it will take longer to do, and you could wear out the EEPROM if you write it over 20,000 times.

Increasing the speed

One thing thats a little annoying about the Arduino Wire library in this case is it is set for 100KHz transfer speed. In the MCP4725 library we update the speed to 400KHz by setting the TWBR

```
TWBR = 12; // 400 khz
```

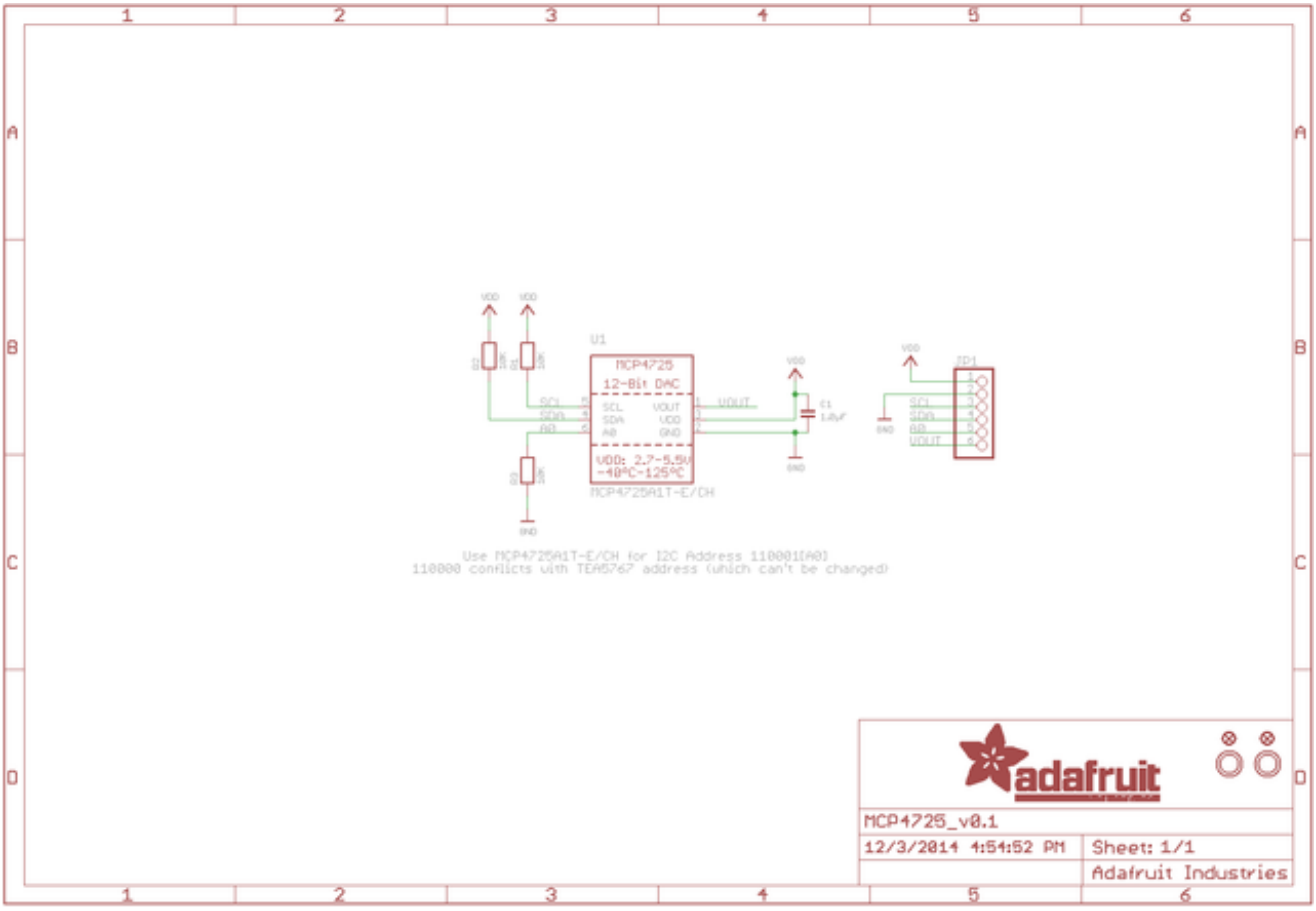
You can speed this up a bit more, if you'd like, check the ATmega328 datasheet for how to calculate the **TWBR** register.

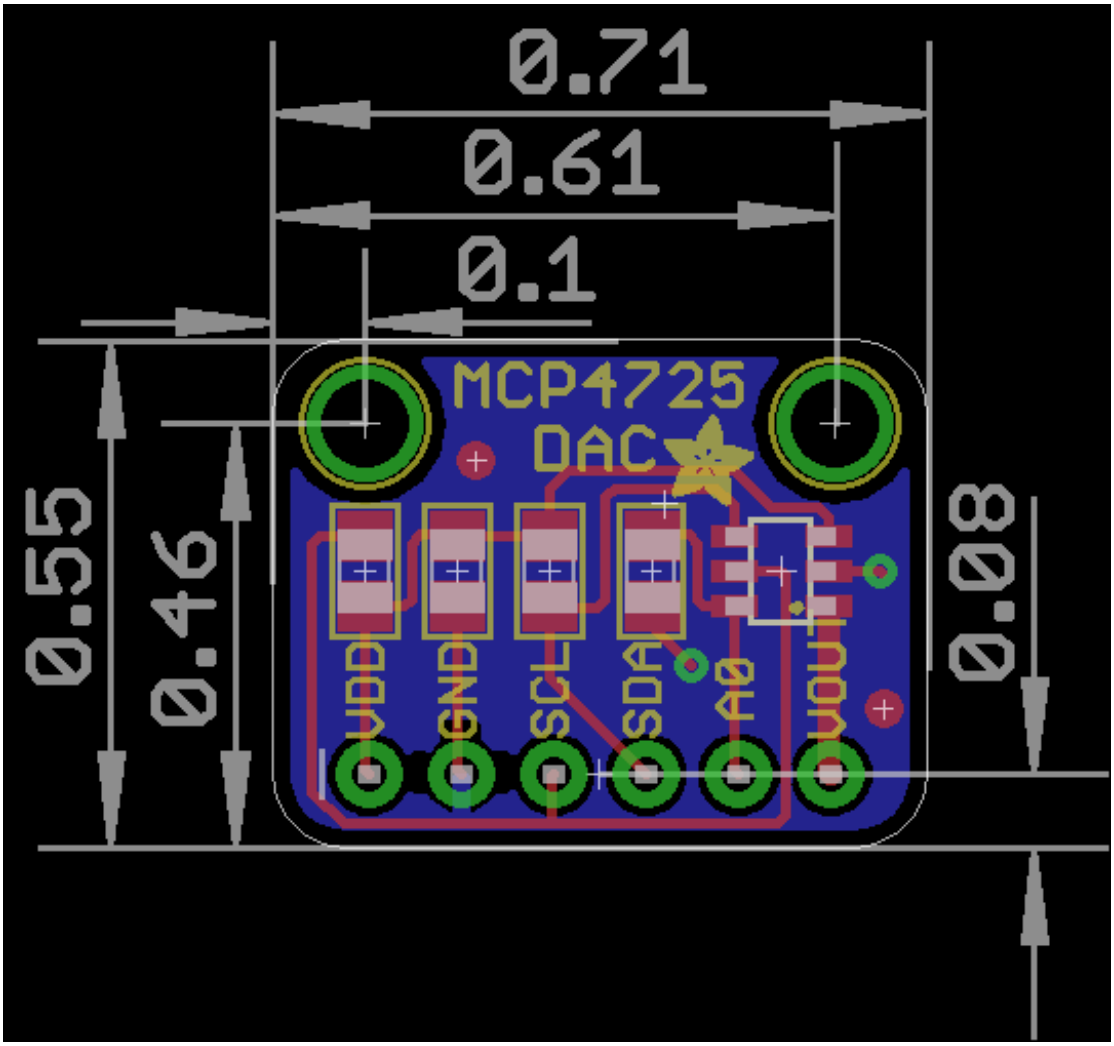
Download

Files

- For more details about the chip, please check out the MCP4725 datasheet (http://adafru.it/aRW)
- MCP4725 Arduino Library is on GitHub (http://adafru.it/aPz)
- Fritzing object in the Adafruit Fritzing library (http://adafru.it/aP3)
- EagleCAD PCB files on GitHub (http://adafru.it/rMC)

Schematic & Fabrication Print




0.71
0.61
0.1
0.55
0.46
0.08
MCP4725
DAC
VDD
GND
SCL
SDA
A0
VOUT