MPL3115A2 Pressure Sensor Hookup Guide

CONTRIBUTORS: *NATE*

MPL3115A2 Overview

The MPL3115A2 is a low-cost, low power, highly accurate barometric
pressure sensor. Use this sensor to detect changes in barometric pressure
(weather changes) or for altitude (UAV controllers and the like). The sensor
is *very* sensitive and capable of detecting a change of only 0.05kPa which
equates to a 0.3m change in altitude.

MPL3115A2 Pressure sensor breakout board

Things you should know about this sensor:

- Uses the I^{2}C interface
- Only one sensor can reside on the I^{2}C bus
- Uses the I^{2}C repeated start condition. Arduino supports this, check if
  you're using a different microcontroller.

- Typical pressure accuracy of ±0.05kPa
- Typical altitude accuracy of ±0.3m
- Typical temperature accuracy of ±3C
- 3.3V sensor - use inline logic level converters or 330 ohm resistors to limit 5V signals
- Here's the datasheet
- Here's the schematic for the breakout board

This sensor is ideal for environmental sensing, a weather station, or datalogging. It is a worthy replacement for the BMP085 and is more sensitive than the MPL115A1.

Suggested Reading

- Read about I^{2}C!
- Using 3.3V sensors with 5V systems
- Installing an Arduino library
- What are pull-up resistors?
- Local Pressure vs Weather Station Pressure

Hooking It Up

Wiring up the MPL3115A2 pressure sensor is very easy!
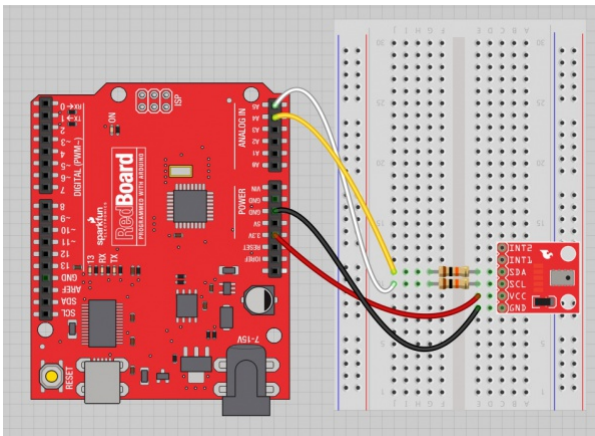
Basic hook up with 330 ohm in-line resistors

There are only four pins that need to be hooked up in order to start using this sensor in a project. One for VCC, one for GND, and two data lines for I^{2}C communication.

Note: This breakout board has built in 1k pull up resistors for I^{2}C communications. If you're hooking up multiple I^{2}C devices on the same bus, you may need to disable the other resistors on the bus.

## Connections: Breakout board to Arduino

- VCC → 3.3V

- GND → GND
- SDA → A4
- SCL → A5

Note: You can also you the SDA and SCL lines on Arduino boards that have them broken out. Make sure you edit the code accordingly if you use those pins instead.

On an Arduino board connect the SDA pin on the breakout board to A4 and SCL to A5. If you're using a 5V Arduino Uno we recommend putting 330 ohm resistors in line to limit the 5V signal going into the sensor and prevent damage to the sensor.
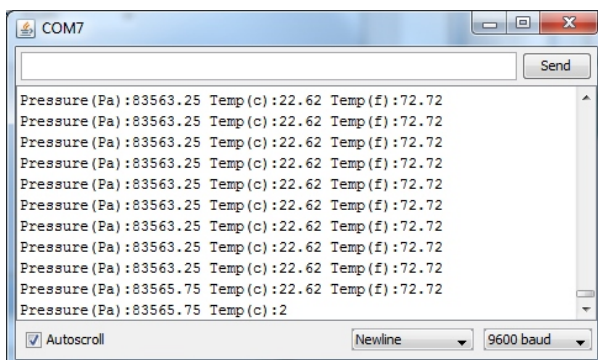
MPL311A2 Library and Functions

We've written an Arduino library and some example code to make using the MPL3115A2 easy to get up and running. Grab the library here. Unzip and paste the folder "MPL3115A2_Pressure" into your Arduino libraries folder. Not sure how to install an Arduino library? Read here for more information.
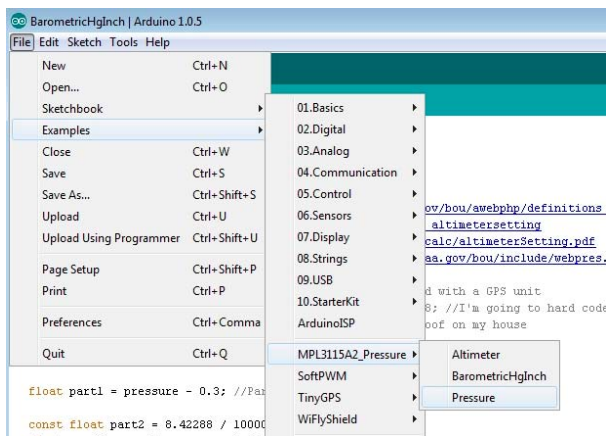
Load the Pressure example

Once the library is installed, open Arduino, and expand the examples menu. You should see the MPL3115A2_Pressure sub-menu. Load the "Pressure" example onto the Arduino. Open the serial terminal at 9600bps. You will see the current barometric pressure and temperature in the room!

Pressure readings!

Explanation of Functions

The library and example code demonstrate the most popular functions supported by the MPL3115A2. Here is an explanation of all the available functions in the library:

- **myPressure.begin()** gets sensor on the I²C bus.
- **myPressure.readAltitude()** returns a float with meters above sea level. Ex: 1638.94
- **myPressure.readAltitudeFt()** returns a float with feet above sea level. Ex: 5376.68
- **myPressure.readPressure()** returns a float with barometric pressure in Pa. Ex: 83351.25
- **myPressure.readTemp()** returns a float with current temperature in Celsius. Ex: 23.37
- **myPressure.readTempF()** returns a float with current temperature in Fahrenheit. Ex: 73.96
- **myPressure.setModeBarometer()** puts the sensor into Pascal measurement mode.
- **myPressure.setModeAltimeter()** puts the sensor into altimetry mode.
- **myPressure.setModeStandy()** puts the sensor into Standby mode. Required when changing CTRL1 register.
- **myPressure.setModeActive()** starts taking measurements!
- **myPressure.setOversampleRate(byte)** sets the # of samples from 1 to 128. See note below *
- **myPressure.enableEventFlags()** sets the fundamental event flags. Required during setup.

When you call the readAltitude, readAltitudeFt, readPressure, or readTemp you will get a float with the sensor reading or an error code:

- 1638.94 is an example of a valid reading.
- -999 indicates that I2C timed out (512ms max). Check your connections.

Table 59. System Output Sample Rate Selection

OS2	OS1	OS0	Oversample Ratio	Minimum Time Between Data Samples
0	0	0	1	6 ms
0	0	1	2	10 ms
0	1	0	4	18 ms
0	1	1	8	34 ms
1	0	0	16	66 ms
1	0	1	32	130 ms
1	1	0	64	258 ms
1	1	1	128	512 ms

Oversample settings

- **setOversampleRate(byte)** receives a value from 0 to 7. Check table 59 above. Allows the user to change sample rate from 1 to 128. Increasing the sample rate significantly decreases the noise of each reading but increases the amount of time to capture each reading. A oversample of 128 will decrease noise to 1.5Pa RMS but requires 512ms per reading. The datasheet recommends oversample of 128 for basic applications.

The MPL3115A2 has a large number of features. Checkout the datasheet for more info. This library covers the fundamentals. Help us out! Please add or suggesting more features on the MPL3115A2 github repo.

Pressure vs Altimeter Setting

If you grabbed a few pressure readings and became confused when you checked your local weather conditions, you're not alone. The absolute pressure that the MPL3115A2 pressure sensor outputs is not the same as what weather stations refer to as pressure. Weather stations report pressure in lots of different units:

- millimeters Mercury (mmHg)
- inches Mercury (inHg)
- millibars or hectopascals (hPa)
- pounds per square inch
- atmospheres (Atm)
- kilogram per centimeter
- inches of water

In barometer mode, the MPL3115A2 outputs pressure readings in Pascals. This is most closely related to millibars or hectopascals. But, why does the sensor not agree with the station around the corner? This is because many stations report pressure in a few different formats. Have a look at all these numbers for the Boulder/Denver area. The key is that your local weather station is probably reporting the *Altimeter setting*.

Thank you National Oceanic and Atmospheric Administration (NOAA)! Did you know they're headquartered here in Boulder, CO?

- **Station pressure** - This is the pressure that is observed at a specific elevation and is the true barometric pressure of a location.
- **Altimeter setting** - This is the pressure reading most commonly heard in radio and television broadcasts. It is not the true barometric pressure at a station. Instead it is the pressure "reduced" to mean sea level using the temperature profile of the "standard" atmosphere, which is representative of average conditions over the United States at 40 degrees north latitude.
- **Mean sea level pressure** - This is the pressure reading most commonly used by meteorologists to track weather systems at the

surface. Like the altimeter setting, it is a "reduced" pressure, which uses observed conditions rather than "standard" conditions to remove the effects of elevation from pressure readings.

The calculation to get from Pascals to 'Altimeter setting' is a bit gnarly:

$$h_m = 0.3048 \times h_{ft}$$
$$Alt = (P_{mb} - 0.3) \times \left(1 + \left(\left(\frac{1013.25^{0.190284} \times 0.0065}{288}\right) \times \left(\frac{h_m}{(P_{mb} - 0.3)^{0.190284}}\right)\right)\right)^{\frac{1}{0.190284}}$$

Formula to convert Pascal pressure to Altimeter setting

Grab the full formula here and give this great Altimeter setting calculator a try. This formula relies on two things: knowing the current pressure in milibars and knowing the height above sea level that the pressure was read. We recommend you capture altitude using a local survey point or a GPS receiver.

If you installed the MPL3115A2 library, you should also have the *BarometricHgInch* example sketch under the Examples->MPL3115A2_Pressure menu under the Arduino IDE. We didn't build this calculation into the library because it could potentially chew up a lot of RAM and code space calculating all the floating point math. But, if you're doing home weather station calculations, this should get you started.

Resources and Going Further

Now that you've got barometric pressure sensing under your belt, consider checking out the following projects and products:

- MPL3115A2 Datasheet
- MPL3115A2 Breakout Schematic
- Combining with an Electric Imp to create a wireless pressure sensor
- Combining multiple pressure sensors to create a tactile robotic hand. Also see TakkTile.
- Checkout the Weather Shield for the MPL3115A2 combined with many other sensors