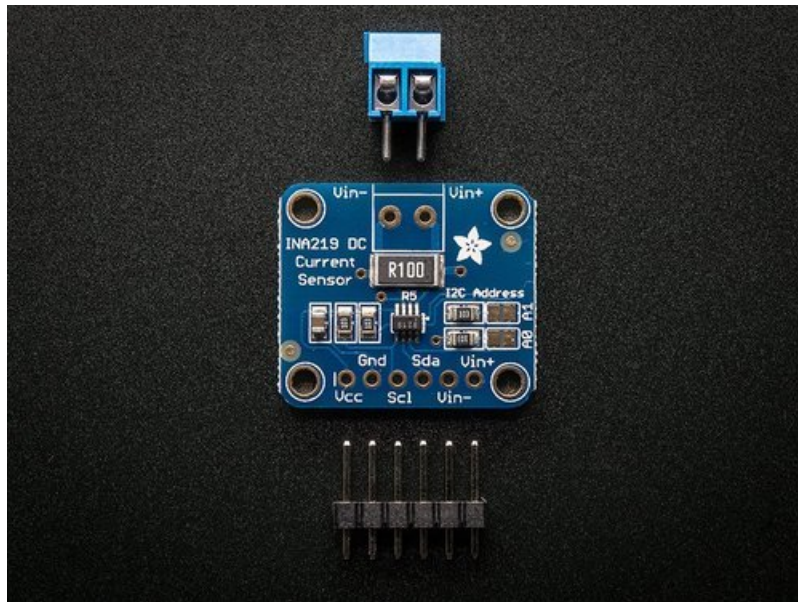




Adafruit INA219 Current Sensor Breakout

Created by lady ada



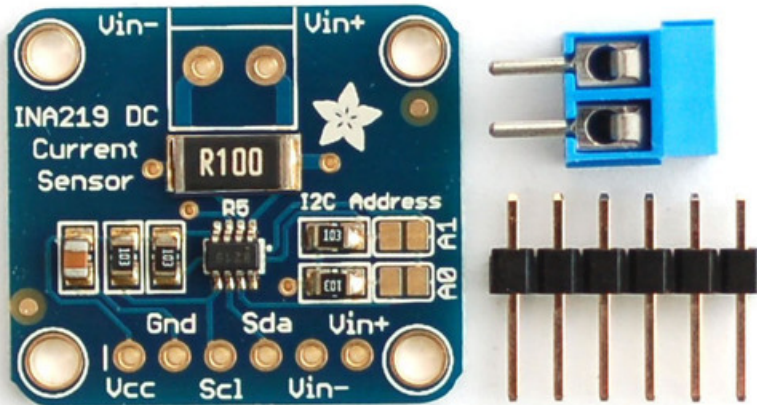
Last updated on 2018-01-17 05:25:30 PM UTC

Guide Contents

Guide Contents	2
Overview	3
Why the High Side?	3
How does it work?	3
Assembly	5
Addressing the Boards	5
Wiring	7
Connect to the microprocessor	7
Connect to the circuit	8
Arduino Code	11
Install the Library	11
Load the Example	11
Run it!	12
Customize it	12
Library Reference	13
Construction and Initialization Functions:	13
Sensor Reading Functions:	13
CircuitPython Code	15
Usage	16
Downloads	18
Datasheets & Files	18
Schematic & Fabrication Print	18

Overview

The INA219B breakout board will solve all your power-monitoring problems. Instead of struggling with two multimeters, you can use this breakout to measure both the high side voltage and DC current draw over I2C with 1% precision.



Why the High Side?

Most current-measuring devices such as our current panel meter are only good for *low side* measuring. That means that unless you want to get a battery involved, you have to stick the measurement resistor between the target ground and true ground.

Since the voltage drop across the resistor is proportional to the current draw, this means that the ground reference will change with varying current. Having a shifting ground reference can cause problems for many circuits.

The INA219B chip is much smarter - it can handle high side current measuring, up to +26VDC, even though it is powered with 3 or 5V. It will also report back that high side voltage, which is great for tracking battery life or solar panels.

How does it work?

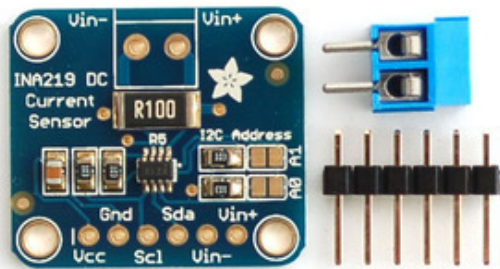
A precision amplifier measures the voltage across the 0.1 ohm, 1% sense resistor. Since the amplifier maximum input difference is $\pm 320\text{mV}$ this means it can measure up to ± 3.2 Amps. With the internal 12 bit ADC, the resolution at $\pm 3.2\text{A}$ range is 0.8mA. With the internal gain set at the minimum of div8, the max current is $\pm 400\text{mA}$ and the resolution is 0.1mA. Advanced hackers can remove the 0.1 ohm current sense resistor and replace it with their own to change the range (say a 0.01 ohm to measure up 32 Amps with a resolution of 8mA)

Note that when switching inductive loads, the instantaneous voltage levels may greatly exceed steady-state levels due to inductive kickback. Chip damage can occur if you do not take precautions to protect against

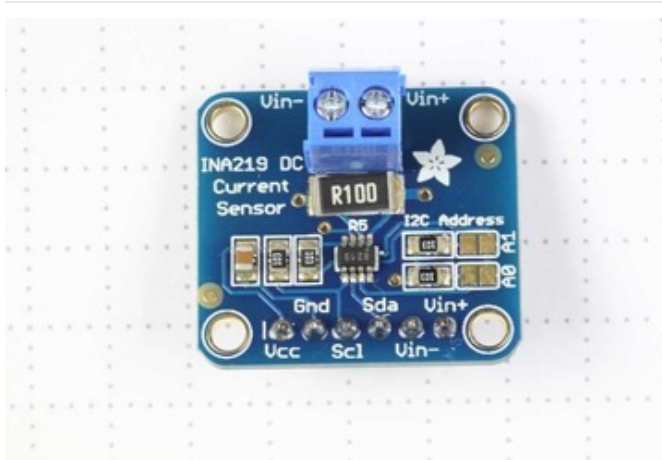
inductive spikes.

Assembly

The board comes with all surface-mount components pre-soldered. Additional parts are included to help integrate the INA219 breakout board into your project.



Wires can be soldered directly to the holes on the edge of the board. But for breadboard use, you will want to solder on the included 6-pin header.

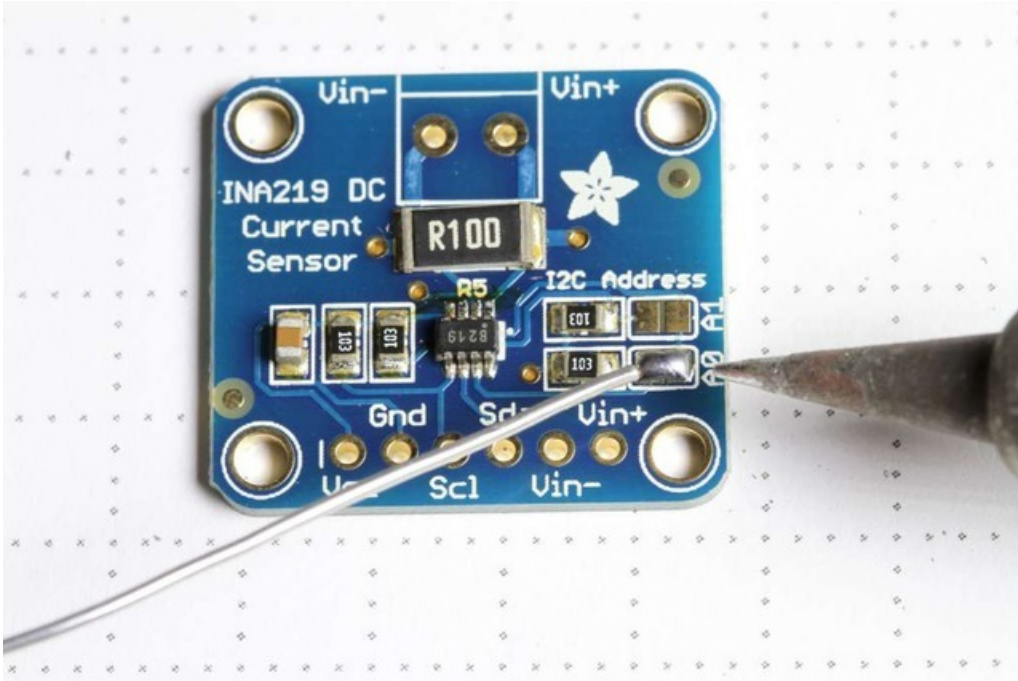


The load can be connected via the header, or using the included 2-pin screw-terminal.

Addressing the Boards

If more than one INA219 breakout board is used, each board must be assigned a unique address. This is done with the address jumpers on the right edge of the board. The I2C base address for each board is 0x40. The binary address that you program with the address jumpers is added to the base I2C address.

To program the address offset, use a drop of solder to bridge the corresponding address jumper for each binary '1' in the address.



Up to 4 boards may be connected. Addressing is as follows:

Board 0: Address = 0x40 Offset = binary 00000 (no jumpers required)

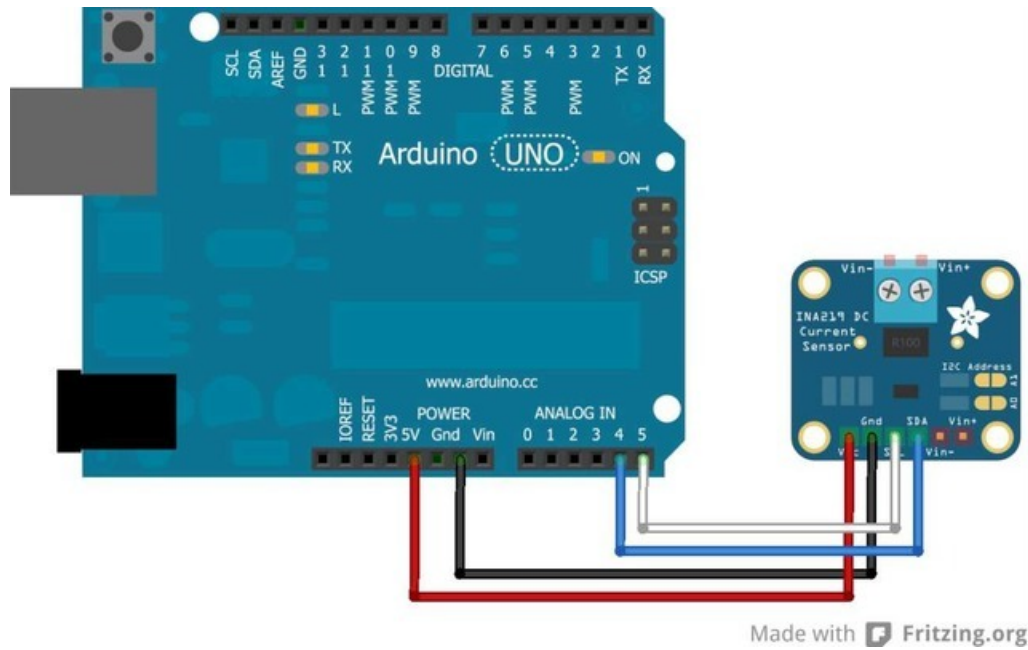
Board 1: Address = 0x41 Offset = binary 00001 (bridge A0 as in the photo above)

Board 2: Address = 0x44 Offset = binary 00100 (bridge A1)

Board 3: Address = 0x45 Offset = binary 00101 (bridge A0 & A1)

Wiring

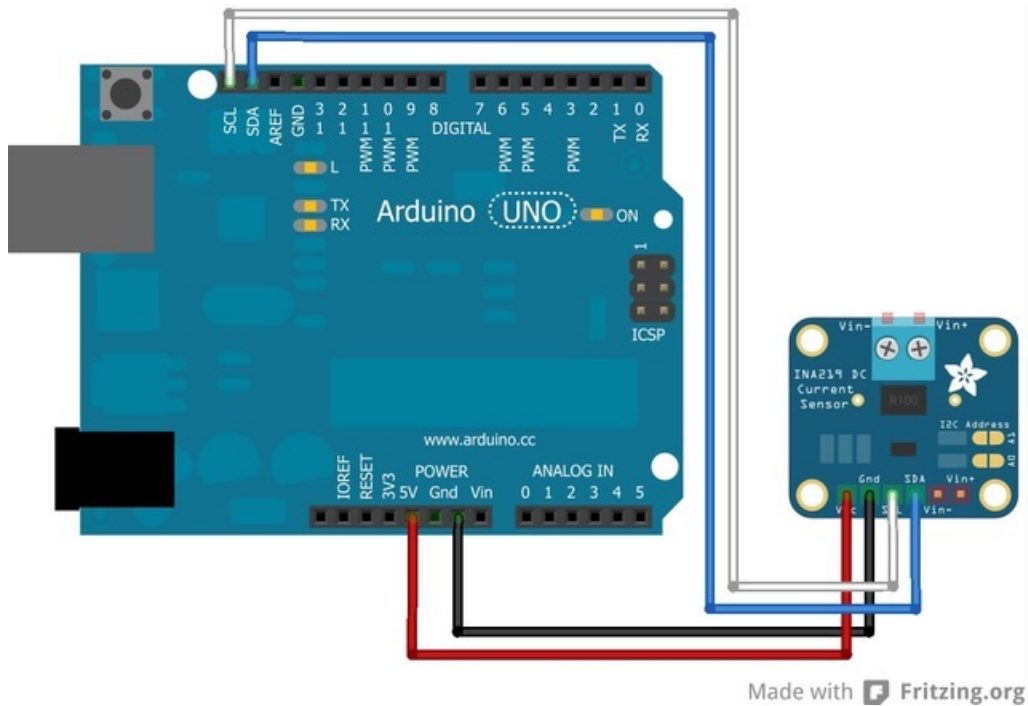
The INA219 breakout board can be powered by the 5v pin on your Arduino and communicates via I2C.



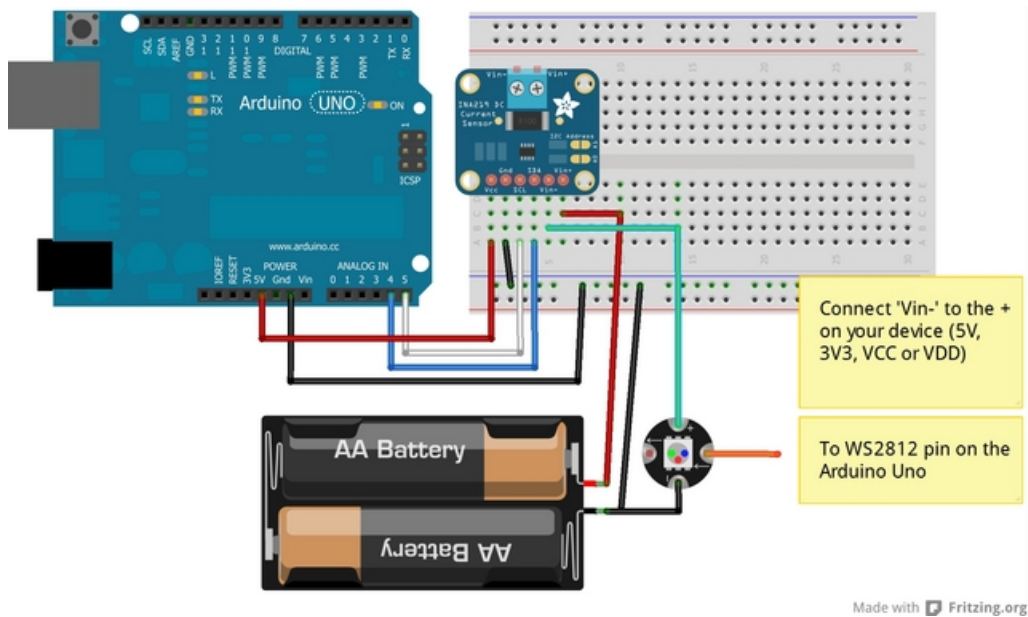
Connect to the microprocessor

- Connect GND to GND
- Connect VCC to 5v
- Then connect SDA to SDA (Analog pin 4 on pre-R3 Arduinos)
- And connect SCL to SCL (Analog pin 5 on pre-R3 Arduinos)

On R3 and later Arduinos, you can connect to the new dedicated SDA & SCL pins next to the AREF pin. On pre-R3 Megas, SDA & SCL are on pins 20 & 21.



Next we must insert the INA219 current sensor into the circuit we want to measure:



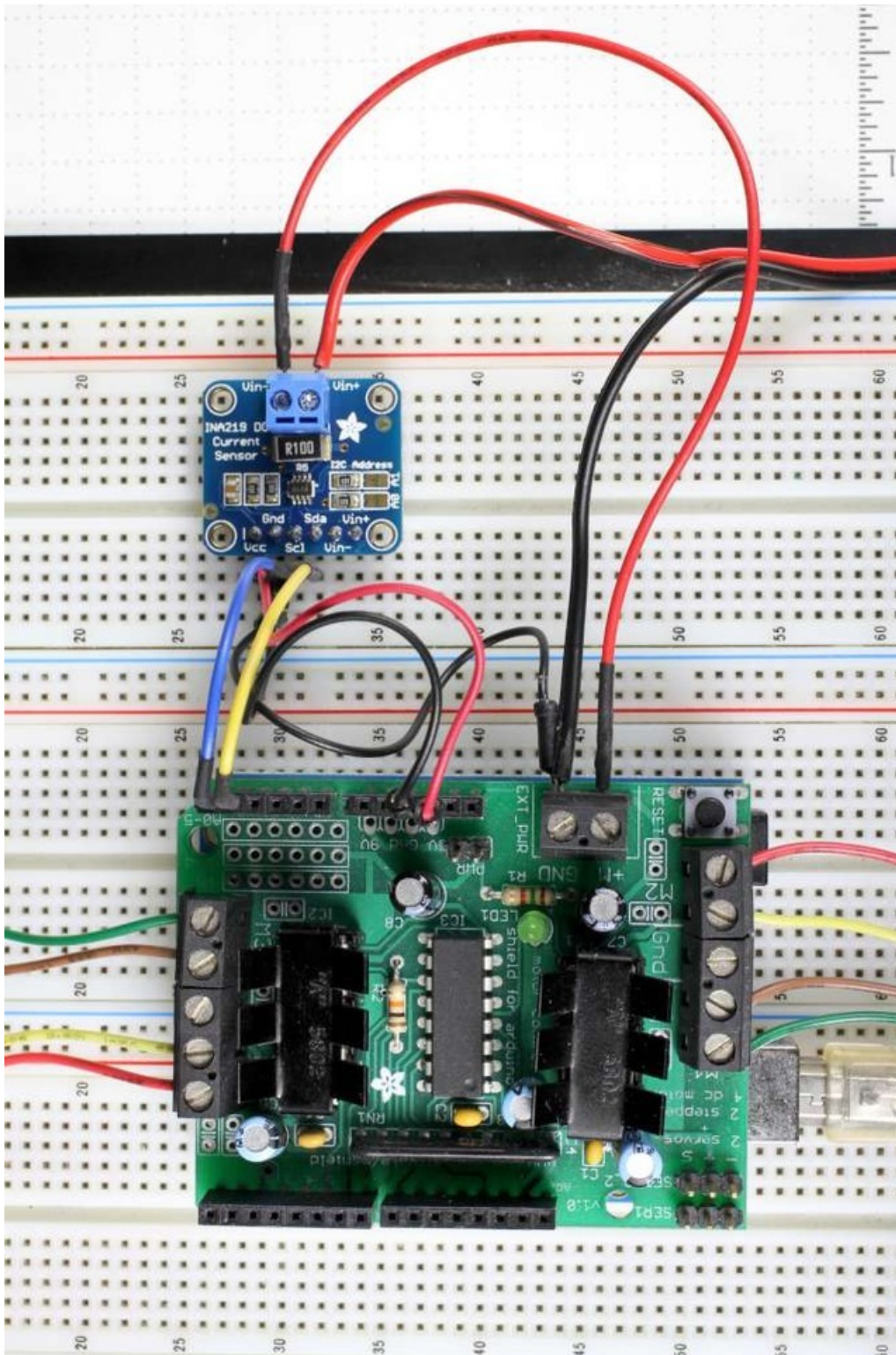
Be careful inserting noisy loads that can cause a sharp current draw, such as DC motors, since they can cause problems on the power lines and may cause the INA219 to reset, etc. When using a DC motor or a similar device, be sure to include a large capacitor to decouple the motor from the power supply and use a snubber diode to protect against inductive spikes.

Connect to the circuit

- Connect V+ to the positive terminal of the power supply for the circuit under test.

- Connect V- to the positive terminal or lead of the load. This puts the sense resistor in-line with the circuit.
- Finally, connect a wire from the negative terminal of the power supply to GND. This allows the sensor to measure the load voltage as well as the load current.

The photo below shows an INA219 breakout board configured to measure the motor supply current on an Adafruit Motor Shield.



Arduino Code

Programming the Adafruit INA219 breakout board is simple using our library:

Install the Library

- [Download the library](#) from the Downloads button below

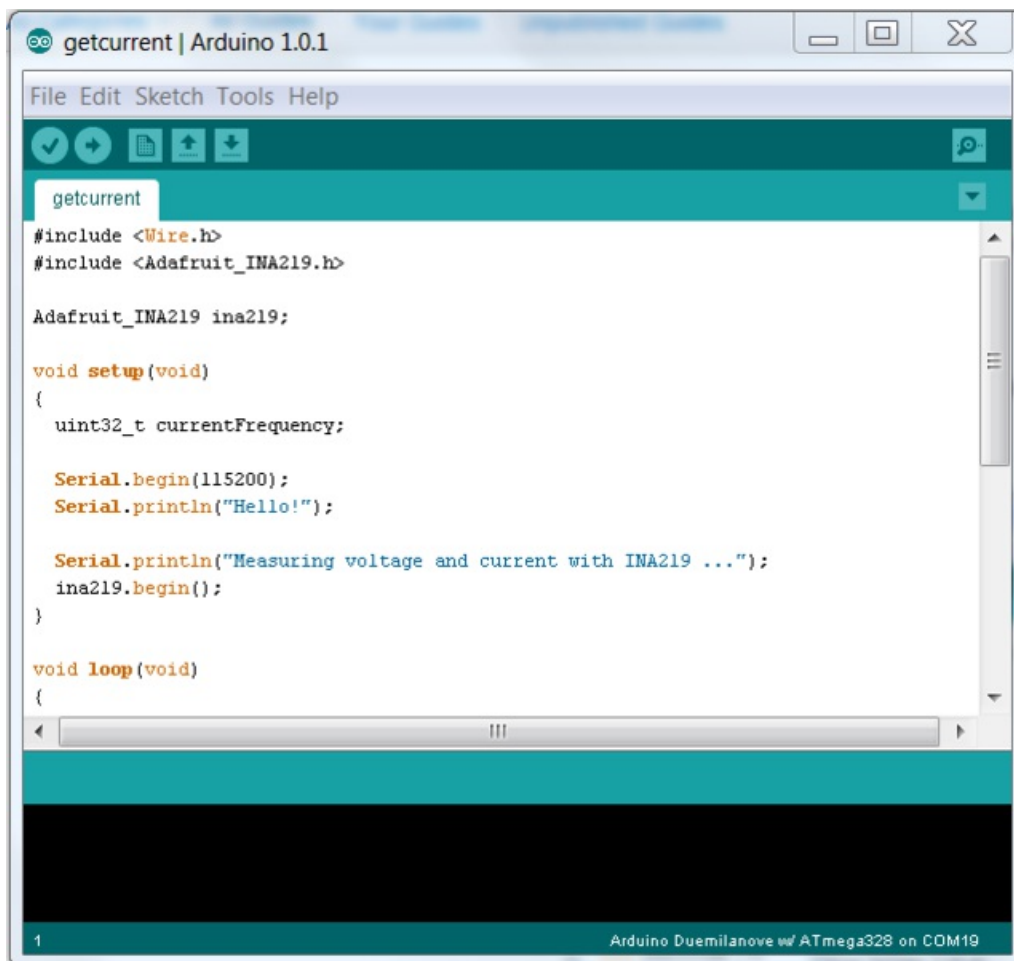
Download Adafruit_INA219

<https://adafru.it/rMD>

- **Expand** the .zip file to the Libraries folder in your Arduino Sketchbook folder (*If you don't know where this is, open File->Preferences in the IDE and it will tell you the location of your sketchbook folder*).
- **Rename** the folder to **Adafruit_INA219**
- **Close** *all* instances of the IDE, then re-open one, so that the IDE will recognize the new library.

Load the Example

- **Select** "File->Examples->Adafruit_INA219->getcurrent"
- The "getcurrent" example code should open in a new IDE window.



```
getcurrent | Arduino 1.0.1
File Edit Sketch Tools Help
getcurrent
#include <Wire.h>
#include <Adafruit_INA219.h>

Adafruit_INA219 ina219;

void setup(void)
{
  uint32_t currentFrequency;

  Serial.begin(115200);
  Serial.println("Hello!");

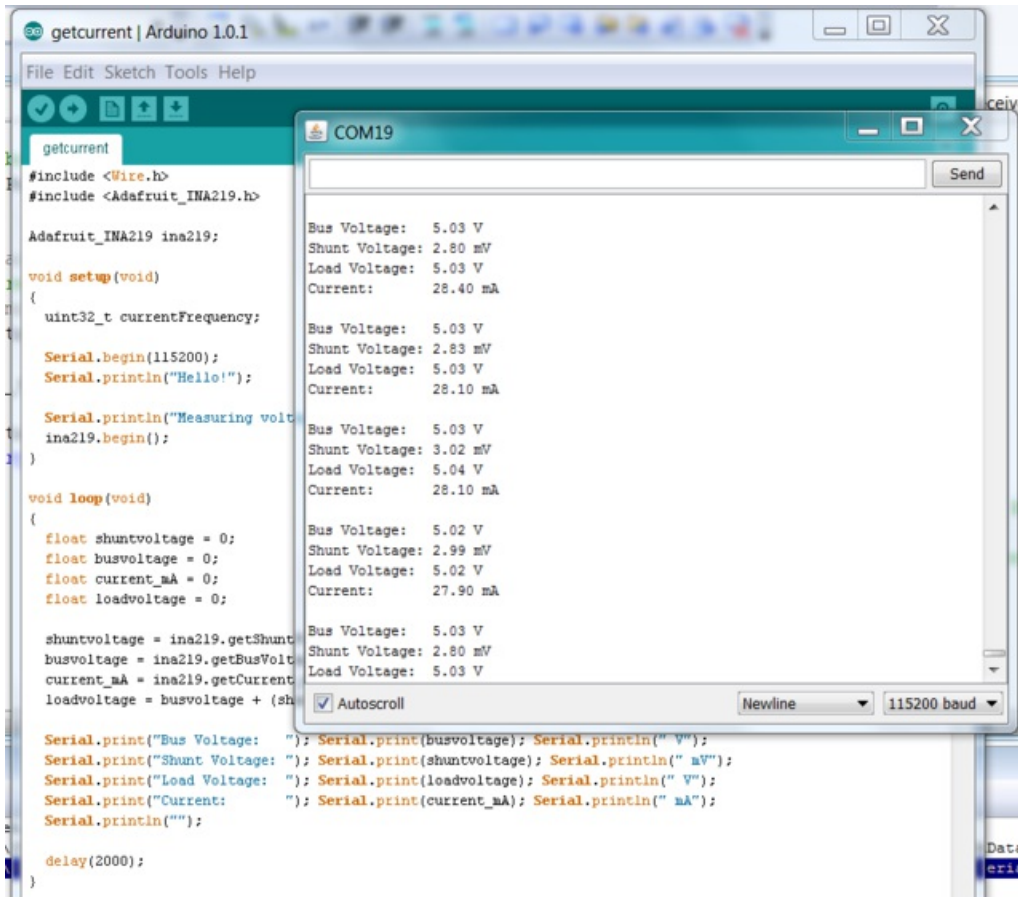
  Serial.println("Measuring voltage and current with INA219 ...");
  ina219.begin();
}

void loop(void)
{
  1
  Arduino Duemilanove w/ ATmega328 on COM19
```

Run it!

- Click on the upload button in the IDE. When it is "done uploading"
- Open your Serial Monitor and set the speed to 115200 baud.

The output should appear similar to the sample below:



The screenshot shows the Arduino IDE interface. The main window displays the 'getcurrent' sketch with the following code:

```
#include <Wire.h>
#include <Adafruit_INA219.h>

Adafruit_INA219 ina219;

void setup(void)
{
  uint32_t currentFrequency;

  Serial.begin(115200);
  Serial.println("Hello!");

  Serial.println("Measuring voltage");
  ina219.begin();
}

void loop(void)
{
  float shuntvoltage = 0;
  float busvoltage = 0;
  float current_mA = 0;
  float loadvoltage = 0;

  shuntvoltage = ina219.getShuntVoltage();
  busvoltage = ina219.getBusVoltage();
  current_mA = ina219.getCurrent_mA();
  loadvoltage = busvoltage + (shuntvoltage * ina219.getShuntResistance());

  Serial.print("Bus Voltage: "); Serial.print(busvoltage); Serial.println(" V");
  Serial.print("Shunt Voltage: "); Serial.print(shuntvoltage); Serial.println(" mV");
  Serial.print("Load Voltage: "); Serial.print(loadvoltage); Serial.println(" V");
  Serial.print("Current: "); Serial.print(current_mA); Serial.println(" mA");
  Serial.println("");

  delay(2000);
}
```

The Serial Monitor window (COM19) shows the following output:

```
Bus Voltage: 5.03 V
Shunt Voltage: 2.80 mV
Load Voltage: 5.03 V
Current: 28.40 mA

Bus Voltage: 5.03 V
Shunt Voltage: 2.83 mV
Load Voltage: 5.03 V
Current: 28.10 mA

Bus Voltage: 5.03 V
Shunt Voltage: 3.02 mV
Load Voltage: 5.04 V
Current: 28.10 mA

Bus Voltage: 5.02 V
Shunt Voltage: 2.99 mV
Load Voltage: 5.02 V
Current: 27.90 mA

Bus Voltage: 5.03 V
Shunt Voltage: 2.80 mV
Load Voltage: 5.03 V
```

The Serial Monitor window also has a 'Send' button, an 'Autoscroll' checkbox, a 'Newline' dropdown menu, and a baud rate dropdown menu set to '115200 baud'.

Customize it

You can adapt, expand or modify the example code to suit your project requirements. For a detailed description of the available library functions, see the **Library Reference** on the next page:

Library Reference

Construction and Initialization Functions:

Adafruit_INA219(uint8_t addr = INA219_ADDRESS);

Constructs an instance of the Adafruit_INA219. If no address is specified, the default address (0x40) is used. If more than one INA219 module is connected, it should be addressed as shown on the Assembly page and the configured address passed to the constructor.

void begin(void);

Initializes I2C communication with the Adafruit_INA219 device using the default configuration values.

Example:

```
#include <Wire.h>
#include <Adafruit_INA219.h>

Adafruit_INA219 ina219_A;
Adafruit_INA219 ina219_B(0x41);

void setup(void)
{
  ina219_A.begin(); // Initialize first board (default address 0x40)
  ina219_B.begin(); // Initialize second board with the address 0x41
}
```

Sensor Reading Functions:

float getBusVoltage_V(void);

Reads the voltage between GND and V-. This is the total voltage seen by the circuit under test. (Supply voltage - shunt voltage).

The return value is in Volts.

float getShuntVoltage_mV(void);

Reads the voltage between V- and V+. This is the measured voltage drop across the shunt resistor.

The return value is in Millivolts.

float getCurrent_mA(void);

Reads the current, derived via Ohms Law from the measured shunt voltage.

The return value is in Milliamps.

Example:

```
float shuntvoltage = 0;
float busvoltage = 0;
float current_mA = 0;
float loadvoltage = 0;

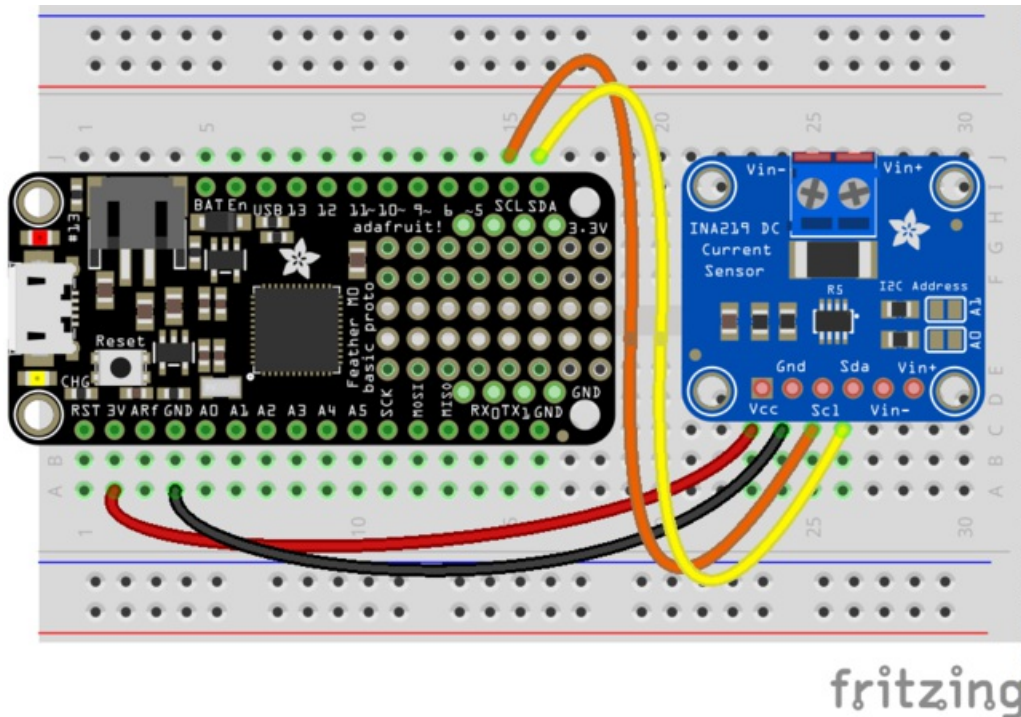
shuntvoltage = ina219.getShuntVoltage_mV();
busvoltage = ina219.getBusVoltage_V();
current_mA = ina219.getCurrent_mA();
loadvoltage = busvoltage + (shuntvoltage / 1000);

Serial.print("Bus Voltage:  "); Serial.print(busvoltage); Serial.println(" V");
Serial.print("Shunt Voltage: "); Serial.print(shuntvoltage); Serial.println(" mV");
Serial.print("Load Voltage:  "); Serial.print(loadvoltage); Serial.println(" V");
Serial.print("Current:      "); Serial.print(current_mA); Serial.println(" mA");
Serial.println("");
```


CircuitPython Code

It's easy to use the INA219 sensor with CircuitPython and the [Adafruit CircuitPython INA219](#) module. This module allows you to easily write Python code that reads the current and more from the sensor.

First wire up a INA219 to your board exactly as shown on the previous pages for Arduino using an I2C interface. Here's an example of wiring a Feather M0 to the sensor with I2C:



- Board 3V to sensor Vcc
- Board GND to sensor GND
- Board SCL to sensor SCL
- Board SDA to sensor SDA

In addition connect some load to measure the current from in series to the sensor's **Vin-** and **Vin+** pins as [mentioned on the wiring page](#).

Next you'll need to install the [Adafruit CircuitPython INA219](#) library on your CircuitPython board. **Remember this module is for Adafruit CircuitPython firmware and not MicroPython.org firmware!**

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](#). For example the Circuit Playground Express 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 Trinket M0, Gemma M0, and Feather/Metro M0 basic you'll need to manually install the necessary libraries from the bundle:

- `adafruit_ina219.mpy`
- `adafruit_bus_device`

You can also download the `adafruit_ina219.mpy` from [its releases page on Github](#).

Before continuing make sure your board's lib folder or root filesystem has the `adafruit_ina219.mpy`, and `adafruit_bus_device` files and folders copied over.

Next [connect to the board's serial REPL](#) so you are at the CircuitPython `>>>` prompt.

Usage

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

```
import board
import busio
import adafruit_ina219
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_ina219.INA219(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_ina219
i2c = bitbangio.I2C(board.SCL, board.SDA)
sensor = adafruit_ina219.INA219(i2c)
```

Now you're ready to read values from the sensor using any of these functions:

- `get_shunt_voltage_mV()` - The shunt voltage in millivolts.
- `get_bus_voltage_V()` - The bus voltage in volts.
- `get_current_mA()` - The current in milliamps.

```
print('Shunt voltage: {} mV'.format(sensor.get_shunt_voltage_mV()))
print('Bus voltage: {} V'.format(sensor.get_bus_voltage_V()))
print('Current: {} mA'.format(sensor.get_current_mA()))
```

```
>>> print('Shunt voltage: {} mV'.format(sensor.get_shunt_voltage_mV()))
Shunt voltage: 7.29 mV
>>> print('Bus voltage: {} V'.format(sensor.get_bus_voltage_V()))
Bus voltage: 4.98 V
>>> print('Current: {} mA'.format(sensor.get_current_mA()))
Current: 74.3 mA
>>> █
```

That's all there is to using the INA219 with CircuitPython!

Here's a full example to print the voltage and current every second. Save this as `main.py` on your board's filesystem and check the output from the serial REPL. Also remember if you're using the ESP8266 you need to change the initialization to use the `bitbangio` module!

```
import board
import busio
import time

import adafruit_ina219

# Initialize I2C bus and sensor.
i2c = busio.I2C(board.SCL, board.SDA)
sensor = adafruit_ina219.INA219(i2c)

# Main loop prints voltage and current every second.
while True:
    print('Shunt voltage: {} mV'.format(sensor.get_shunt_voltage_mV()))
    print('Bus voltage: {} V'.format(sensor.get_bus_voltage_V()))
    print('Current: {} mA'.format(sensor.get_current_mA()))
    time.sleep(1.0)
```

Downloads

Datasheets & Files

- [Eagle PCB files for the INA219 breakout board](#)
- [Arduino driver library](#)
- [Data Sheet for the INA219 chip](#)
- [Fritzing object in the Adafruit Fritzing library](#)

Schematic & Fabrication Print

