Adafruit LC709203F LiPoly / Lilon Fuel Gauge and Battery Monitor

Created by Kattni Rembor

https://learn.adafruit.com/adafruit-lc709203f-lipo-lipoly-battery-monitor

Last updated on 2023-08-29 04:30:42 PM EDT
# Table of Contents

## Overview 3

## Pinouts 5
- Power Pins
- I2C Logic Pins
- JST Ports
- Other Pins
- LED Trace on the Back of the Breakout

## Arduino Use 7
- I2C Wiring
- Library Installation
- Load Example

## Arduino Docs 10

## Python & CircuitPython 10
- CircuitPython Microcontroller Wiring
- Python Computer Wiring
- CircuitPython Installation of LC709203F Library
- Python Installation of LC709203F Library
- CircuitPython & Python Usage
- Full Example Code

## WipperSnapper Setup 14
- What is WipperSnapper
- Wiring
- Usage

## Downloads 21
- Files:
- 3D Model
- Schematic
- Fab Print
Overview

Low cost Lithium Polymer batteries have revolutionized electronics - they're thin, they're light, they can be regulated down to 3.3V and they're easy to charge. On your phone, there's a little image of a battery cell that tells you the percentage of charge - so you know when you absolutely need to plug it in and when you can stay untethered. The Adafruit LC709203F LiPoly / Lilon Fuel Gauge and Battery Monitor does the same thing. Connect it to your Lipoly or Lilon battery and it will let you know the voltage of the cell, it does the annoying math of decoding the non-linear voltage to get you a valid percentage as well!
Since this nice chip is I2C, it works with any and all microcontroller or microcomputer boards, from the Arduino UNO up to the Raspberry Pi. And you don't have to worry about logic level, as the gauge runs with 3.3V or 5.0V power and logic equally fine.

To use, connect the single-cell battery to one of the JST 2 PH ports (either one). Then use the included JST PH jumper cable to connect to your boost converter, Feather, whatever! Use the I2C interface and our Arduino or CircuitPython/Python library code to set the pack size (this helps tune the calculation) and read the voltage and percentage whenever you like. If you connect a 10K thermistor to the THERM pin you can also use it to read the battery pack temperature - our packs do not come with a built in thermistor but many do.
To get you going fast, we spun up a custom made PCB in the **STEMMA QT form factor** (), making it easy to interface with. The **STEMMA QT connectors** () on either side are compatible with the **SparkFun Qwiic** () I2C connectors. This allows you to make solderless connections between your development board and the LC709203 or to chain it with a wide range of other sensors and accessories using a **compatible cable** ()

---

### Pinouts

[Image of the LC709203 pinout diagram]
Power Pins

- **VIN** - The chip can safely run from 3-5VDC. To power the board, give it the same power as the logic level of your microcontroller - e.g. for a 5V microcontroller like Arduino, use 5V
- **GND** - common ground for power and logic

The LC709203F is powered by the connected battery, not by VIN or the STEamma/QT connector. If no battery is plugged in, or the battery is too low, the LC709203F will not respond to I2C scans or commands.

I2C Logic Pins

- **SCL** - This is the I2C clock pin SCL, connect to your microcontroller's I2C clock line. There's a 10K pullup on this pin.
- **SDI** - This is the I2C data pin SDA, connect to your microcontroller's I2C data line. There's a 10K pullup on this pin.
- **STEMMA QT ()** - These connectors allow you to connect to dev boards with STE MMA QT connectors or to other things with various associated accessories()

JST Ports

There are two JST ports, they are equivalent. Connect the battery to either one, then the load/charger to the other. Watch out for battery polarity if not using an Adafruit battery!

The two ports simply connect together, and to the battery. You can use/charge the battery while connected by having the battery on one port, and then connecting a charger/project to the other

Other Pins

- **INT** - Interrupt signal out, you can set this up to pull low when the voltage or percentage drops below a threshold. Pulled up to VIN with a 10K resistor
- **Therm** - Connect a 10K thermistor to the THERM pin to read the battery pack temperature - our packs do not come with a built in thermistor but many do. The other side of the thermistor goes to ground.
LED Trace on the Back of the Breakout

There are two pads connected by a trace on the back of the board, labeled LED. If you want to conserve power when running only on battery, you can cut this trace to disable the LED on the breakout.

Arduino Use

Using the LC709203F with Arduino is a simple matter of wiring up the sensor to your Arduino-compatible microcontroller, installing the Adafruit LC709203F () library we’ve written, and running the provided example code.

I2C Wiring

Use this wiring to connect via I2C interface. The LC709203F uses I2C address 0x0B and it cannot be changed.

Here is how to wire up the sensor using one of the STEemma QT () connectors. The examples show a Metro but wiring will work the same for an Arduino or other compatible board.

Connect board VIN (red wire) to Arduino 5V if you are running a 5V board Arduino (Uno, etc.). If your board is 3V, connect to that instead.
Connect board GND (black wire) to Arduino GND
Connect board SCL (yellow wire) to Arduino SCL
Connect board SDA (blue wire) to Arduino SDA
Plug a 3.7/4.2V lithium polymer or lithium ion rechargeable battery into either of the JST battery ports.

Watch out for battery polarity! A reversed battery will damage the monitor. There are + and - symbols on the PCB to indicate which is which.
Library Installation

You can install the Adafruit LC709203F () library for Arduino using the Library Manager in the Arduino IDE.

Click the Manage Libraries ... menu item, search for Adafruit LC709203F, and select the Adafruit LC709203F library.

Follow the same process for the Adafruit BusIO library.

Load Example

Open up File -> Examples -> Adafruit LC709203F -> LC709203F_demo

After opening the demo file, upload to your Arduino wired up to the battery breakout.

Make sure a battery is plugged into the board, it must be powered by the battery to function!
Once you upload the code, you will see the Battery Voltage, Charge Percentage, and Temperature values being printed when you open the Serial Monitor (Tools->Serial Monitor) at 115200 baud, similar to this:

![Image of serial monitor output]

The battery temperature will not be accurate UNLESS you have connected a 10K ohm thermistor to the THERM input pin! If you don't have the thermistor attached, ignore the temp reading!

The LC chip works best when queried every few seconds at the fastest. Don't disconnect the LiPo battery, it is used to power the LC chip!

You can tweak the calculation for the battery pack size, by changing the line `setPackSize(LC709203F_APA_500MAH);`

Pick the closest of the following values for your battery size:

- LC709203F_APA_100MAH
- LC709203F_APA_200MAH
- LC709203F_APA_500MAH
- LC709203F_APA_1000MAH
- LC709203F_APA_2000MAH
- LC709203F_APA_3000MAH

You can set an alarm for voltage or percent with `setAlarmRSOC (percent)` or `setAlarmVoltage (floating point voltage)` (), and check the INT pin going low.
Python & CircuitPython

It's easy to use the LC709203F and the Adafruit CircuitPython LC709203F module. This library allows you to easily write Python code that provides information about the connected battery including voltage, charge percentage, and temperature.

You can use this sensor with any CircuitPython microcontroller board or with a computer that has GPIO and Python thanks to Adafruit_Blinka, our CircuitPython-for-Python compatibility library.

CircuitPython Microcontroller Wiring

Wire up a LC709203F to your board and a battery exactly as shown below. Here's an example of wiring a Feather M4 to the sensor with I2C using STEMMA QT and a solderless breadboard.

Connect board VIN (red wire) to Feather 3V
Connect board GND (black wire) to Feather GND
Connect board SCL (yellow wire) to Feather SCL
Connect board SDA (blue wire) to Feather SDA
Plug a 3.7/4.2V lithium polymer or lithium ion rechargeable battery into either of the JST battery ports on the board.
Plug the other board JST Battery port into the Feather JST port using the cable included with the board.
Python Computer Wiring

Since there's 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.

Here's the Raspberry Pi wired to the sensor with I2C using STEMMA QT and a solderless breadboard.

Watch out for battery polarity! A reversed battery will damage the monitor. There are + and - symbols on the PCB to indicate which is which.

CircuitPython Installation of LC709203F Library

You'll need to install the Adafruit CircuitPython LC709203F 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 CircuitPython starter guide has a great page on how to install libraries from the bundle.

Load the the following libraries into the lib folder on your CIRCUITPY drive:

- adafruit_lc709203f.mpy
- adafruit_bus_device

Before continuing make sure your board’s lib folder or root filesystem has the adafruit_lc709203f.mpy file and adafruit_bus_device folder copied over.

Next connect to the board’s serial console so you are ready to see the example output.

On the ESP32-S3 only, there is a problem with the I2C implementation in CircuitPython that causes the LC709203F to be unreadable. This issue is being tracked at https://github.com/adafruit/circuitpython/issues/6311.

Python Installation of LC709203F 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!

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

```
pip3 install adafruit-circuitpython-lc709203f
```

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 & Python Usage

To demonstrate the usage of the sensor we'll initialize it and read the battery voltage and percentage from the board's Python REPL.

Run the following code to import the necessary modules and initialize the I2C connection with the sensor:

```python
import board
from adafruit_lc709203f import LC709203F
sensor = LC709203F(board.I2C())
```

Now you're ready to use the following properties:

- `ic_version` - Read-only chip version
- `cell_voltage` - Floating point voltage
- `cell_percent` - Percentage of cell capacity
- `power_mode` - Current power mode (operating or sleeping)
- `pack_size` - current battery pack size

For example to print the board version, and battery voltage and percentage values:

```python
print("IC version:", hex(sensor.ic_version))
print("Battery voltage: %0.3f Volts" % (sensor.cell_voltage))
print("Battery percentage: %0.1f %" % (sensor.cell_percent))
```

That's all there is to using the LC709203F LiPoly / Lilon Fuel Gauge and Battery Monitor with CircuitPython!
Full Example Code

```python
import time
import board
from adafruit_lc709203f import LC709203F

print("LC709203F simple test")
print("Make sure LiPoly battery is plugged into the board!")

i2c = board.I2C()  # uses board.SCL and board.SDA
sensor = LC709203F(i2c)

print("IC version: ", hex(sensor.ic_version))

while True:
    try:
        print("Battery: %0.3f Volts / %0.1f %%")
        print("% (sensor.cell_voltage, sensor.cell_percent)"
    except OSError:
        print("retry reads")

time.sleep(1)
```

WipperSnapper Setup

To use the LC709203F with WipperSnapper, you must be running WipperSnapper Beta 45 or newer. Better yet, install the latest WipperSnapper version.

What is WipperSnapper

WipperSnapper is a firmware designed to turn any WiFi-capable board into an Internet-of-Things device without programming a single line of code. WipperSnapper connects to Adafruit IO, a web platform designed (by Adafruit!) to display, respond, and interact with your project's data.

Simply load the WipperSnapper firmware onto your board, add credentials, and plug it into power. Your board will automatically register itself with your Adafruit IO account.

From there, you can add components to your board such as buttons, switches, potentiometers, sensors, and more! Components are dynamically added to hardware,
so you can immediately start interacting, logging, and streaming the data your projects produce without writing code.

If you've never used WipperSnapper, click below to read through the quick start guide before continuing.

Quickstart: Adafruit IO WipperSnapper

Wiring

First, wire up an LC709203F to your board exactly as follows. Here is an example of the LC709203F wired to an Adafruit ESP32 Feather V2 using I2C with a STEMMA QT cable (no soldering required)

Board 3V to sensor VIN (red wire on STEMMA QT)
Board GND to sensor GND (black wire on STEMMA QT)
Board SCL to sensor SCL (yellow wire on STEMMA QT)
Board SDA to sensor SDA (blue wire on STEMMA QT)

Usage

Connect your board to Adafruit IO Wippersnapper and navigate to the WipperSnapper board list.
On this page, select the WipperSnapper board you're using to be brought to the board's interface page.

If you do not see your board listed here - you need to connect your board to Adafruit IO first.

On the device page, quickly check that you're running the latest version of the WipperSnapper firmware.

The device tile on the left indicates the version number of the firmware running on the connected board.

If the firmware version is green with a checkmark - continue with this guide.
If the firmware version is red with an "X" - update to the latest WipperSnapper firmware on your board before continuing.

Next, make sure the sensor is plugged into your board and click the I2C Scan button.
You should see the LC709203F's default I2C address of 0x0b pop-up in the I2C scan list.

I2C Scan Complete

|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 00|   |   |   |   |   |   |   |   |   |   | 0b|   |   |   |   |   |   |   |   |   |   |
| 10|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 20|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 30|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 40|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 50|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 60|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
| 70|   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |

I don't see the sensor's I2C address listed!

First, double-check the connection and/or wiring between the sensor and the board.

Then, reset the board and let it re-connect to Adafruit IO WipperSnapper.

With the sensor detected in an I2C scan, you're ready to add the sensor to your board.

Click the New Component button or the + button to bring up the component picker.
On the component configuration page, the LC709203F's sensor address should be listed along with the sensor's settings.

The Send Every option is specific to each measurement. This option will tell the Feather how often it should read from each of the LC709203F's two measurements and send the data to Adafruit IO. Measurements can range from every 30 seconds to every 24 hours.

For this example, set the Send Every interval for each sensor to every 30 seconds.
Your device interface should now show the sensor components you created. After the interval you configured elapses, WipperSnapper will automatically read values from the sensor(s) and send them to Adafruit IO.

To view the data that has been logged from the sensor, click on the graph next to the sensor name.
Here you can see the feed history and edit things about the feed such as the name, privacy, webhooks associated with the feed and more. If you want to learn more about how feeds work, check out this page.

The LC709203F has two measurements that each have their own feeds. In this picture, we're looking at the battery cell percent reading, but if you click on the graph icon for the other measurement you'll see its feed history.

For IO Free accounts, feed data is stored for a maximum of 30 days and there's a maximum of 10 feeds. In this guide, you created two feeds (one for each of the LC709203F's readings). If you'd like to store data for more than 30 days, increase the number of feeds (components) you can use with WipperSnapper, or increase your data rate to send more sensor measurements to Adafruit IO - upgrade your account to Adafruit IO Plus.
Downloads

Files:

- LC709203F datasheet
- Fritzing object in the Adafruit Fritzing Library
- EagleCAD PCB files on GitHub
- 3D Models on GitHub

3D Model