

# Gravity: 3.7V Li Battery Fuel Gauge SKU: DFR0563

---



## Introduction

3.7V lithium batteries are commonly used in many projects. However, The remaining power and voltage of the lithium battery are highly nonlinear. We can only roughly judge whether the battery is full or almost empty according to its voltage. The Gravity 3.7V Lithium Battery Fuel Gauge employs Gravity I2C interface, ultra-low operating current, and real-time tracking of the relative state of charge (SOC) of the battery through Maxim's patented algorithm, eliminating the need for full-to-empty relearning and offset accumulation errors. Plug and play to accurately measure the voltage and remaining power of the battery. The module also features as a low battery power alert interrupt function. When the battery power falls below specified threshold, the ALR pin generates a falling pulse to trigger the external interrupt of the controller.

You will find it a great help to estimate the battery life by learning the power consumption of the system with this module. In the solar powered projects, it helps to record the status of the battery power change in a day or even months, which benefits to understand the power balance of charge and discharge of the solar system.

# Features

- Maxim's patented algorithm, accurate voltage and remaining power readings of lithium battery
- No charge-discharge relearning process, no offset accumulation errors, plug and play
- Wide input voltage, compatible with 3.3V and 5V controllers
- Programmable low power alert interrupt threshold
- Battery reverse connection protection

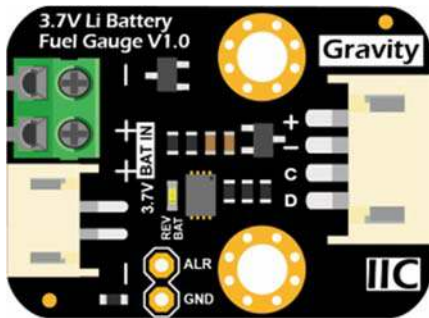
# Specifications

- Input Voltage (VCC): 3.3V~6.0V
- Battery Input Voltage (BAT IN): 2.5V~4.2V
- Battery Type(BAT IN): 3.7V Li-polymer/Li-ion battery
- Operating Current: 50 uA
- Interface: Gravity I2C (logic level: 0-3.3V)
- Dimension: 22.0mm\*30.0mm

# Applications

- Solar Weather Station, Street Light
- Lithium Battery Charger
- Robotics

# Board Overview



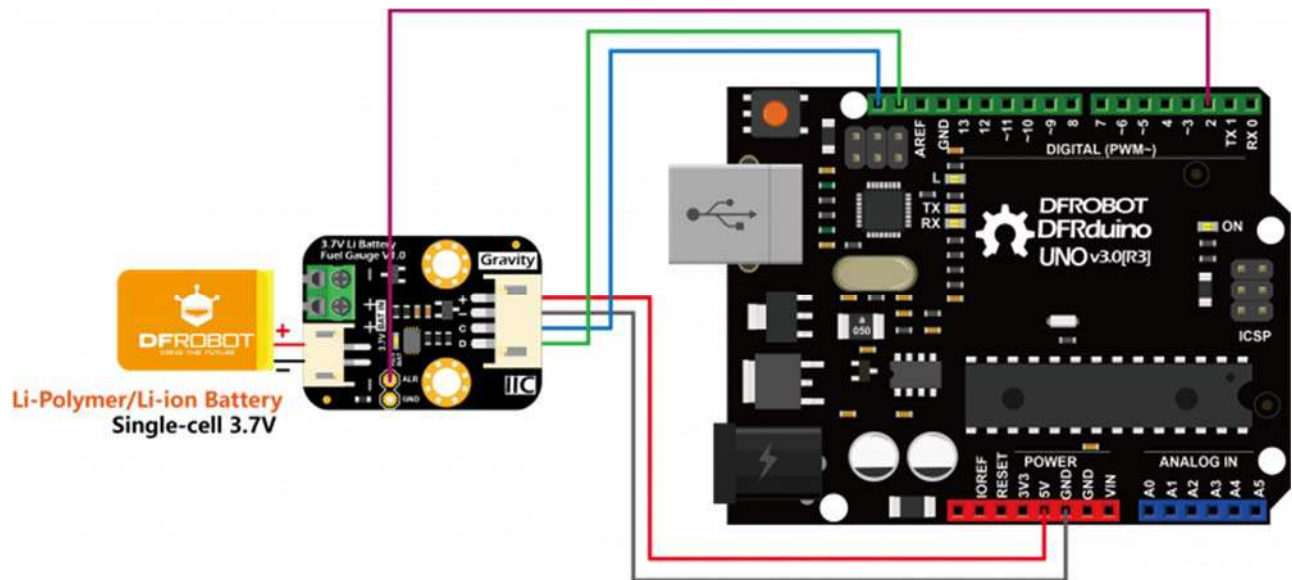
Label	Name	Description
+	VCC	Power VCC (3.3~6.0V)
-	GND	Power GND
C	SCL	I2C Clock Signal
D	SDA	I2C Data Signal
ALR	ALRT*	Low battery power alert interrupt (active low)
BAT IN	Battery input	3.7V lithium battery connection port
REV BAT	Battery reverse Indicator	When the battery is reversed, the reverse polarity protection circuit is activated and REV BAT lights up.

# Arduino Tutorial

## Requirements

- **Hardware**
- DFRduino UNO R3 (or similar) x 1
- DFRobot Gravity: 3.7V Li Battery Fuel Gauge x 1
- Gravity-4P sensor wire (or Dupont wires) x 1
- **Software**
- Arduino IDE (V1.0.x or V1.8.x), [Click to Download Arduino IDE from Arduino®](#)
- Download and install the [DFRobot\\_MAX17043 Library](#). [How to install the library?](#)

## Connection Diagram



## Read Battery Voltage, Remaining Power and Set Low Power Interrupt Alert

- Connect the module to the Arduino according to the connection diagram. The battery can be connected to the 2P terminal or JST-PH2.0 2P connector. These two connectors are connected internally in parallel. The I2C address is fixed to 0x36.
- Install DFRobot\_MAX17043 library.
- Open Arduino IDE, upload the following sample code to the Arduino UNO.
- Open the serial monitor of the Arduino IDE and set the baud rate to 115200. The module prints the battery voltage, the percentage of remaining battery power, and the interrupt alert message through the serial port.

```

/*
 * file DFRobot_MAX17043.ino
 *
 * connect gauge I2C interface with your board (please reference board compat
ibility)
 *
 * Voltage, percentage will be printed via serial.
 * Use API to config alaram and sleep (please reference to the readme in lib)
 *
 * Copyright [DFRobot] (http://www.dfrobot.com), 2016
 * Copyright GNU Lesser General Public License
 *
 * version V1.0
 * date 2018-3-26
 */

#include "DFRobot_MAX17043.h"
#include "Wire.h"

#ifdef __AVR__
  #define ALR_PIN 2
#else
  #define ALR_PIN D2
#endif

#define PRINT_INTERVAL 2000
#define BAT_PERCENTAGE 32

DFRobot_MAX17043 gauge;
uint8_t intFlag = 0;

void interruptCallBack()
{
  intFlag = 1;
}

```

```

}

void setup()
{
  Serial.begin(115200);
  while(!Serial);
  Serial.println();
  Serial.println();
  pinMode(ALR_PIN, INPUT_PULLUP);
  attachInterrupt(digitalPinToInterrupt(ALR_PIN), interruptCallback, FALLING)
; //default alert is 32%

  while(gauge.begin() != 0) {
    Serial.println("gauge begin failed!");
    delay(2000);
  }
  delay(2);
  Serial.println("gauge begin successful!");

  //gauge.setInterrupt(BAT_PERCENTAGE); //use this to modify alert threshold
as 1% - 32% (integer)
}

void loop()
{
  static unsigned long lastMillis = 0;
  if((millis() - lastMillis) > PRINT_INTERVAL) {
    lastMillis = millis();
    Serial.println();

    Serial.print("voltage: ");
    Serial.print(gauge.readVoltage());
    Serial.println(" mV");

    Serial.print("percentage: ");
    Serial.print(gauge.readPercentage());

```

```

    Serial.println(" %");
}

if(intFlag == 1) {
    intFlag = 0;
    gauge.clearInterrupt();
    Serial.println("Low power alert interrupt!");
    //put your battery low power alert interrupt service routine here
}
}

```

## Results

- Arduino prints the current voltage (voltage), remaining power (percentage), and interrupt alert (if any) information every 2 seconds.
- Battery low power interrupt alert ALR instructions:
- The default value of battery low power interrupt alert threshold is 32%. That is, when the remaining power is lower than 32%, a falling edge interrupt is generated on the ALR pin. The threshold BAT\_PERCENTAGE can be set to any integer between 1-32 (corresponding to 1%-32%, respectively). This threshold can be set by the function setInterrupt().
- When the battery's initial remaining power is higher than BAT\_PERCENTAGE, the ALR pin is set high. If it falls below BAT\_PERCENTAGE (due to discharge) , ALR is pulled to low. The controller is triggered to print "Low power alert interrupt!", and clear interrupt through clearInterrupt(), which causes ALR back to high immediately.
- When the battery's initial remaining power is below BAT\_PERCENTAGE, ALR generates a interrupt at the beginning.
- After the battery remaining power grows higher than BAT\_PERCENTAGE (due to charge) , another interrupt will be generated when the power again falls below BAT\_PERCENTAGE (due to discharge). If the clearInterrupt() is not called after the interrupt is occurred, ALR remains low regardless of the battery remaining power.

```
gauge begin successful!  
Low power alert interrupt!
```

```
voltage: 3675.00 mV  
percentage: 7.60 %
```

```
voltage: 3675.00 mV  
percentage: 7.60 %
```

```
voltage: 3676.25 mV  
percentage: 7.60 %
```

```
voltage: 3676.25 mV  
percentage: 7.60 %
```

```
voltage: 3676.25 mV  
percentage: 7.60 %
```

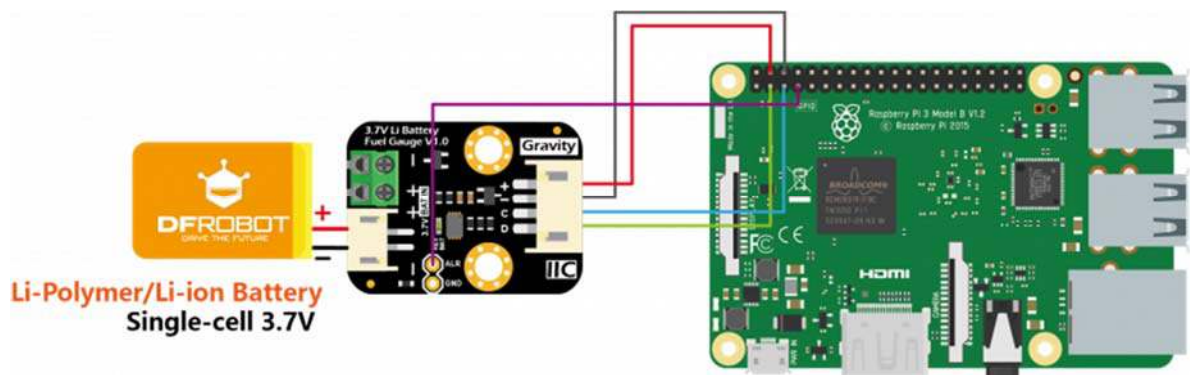
---

# Raspberry Pi Tutorial

## Requirements

- **Hardware**
- Raspberry Pi 3 Model B (or similar) x 1
- DFRobot Gravity: 3.7V Li Battery Fuel Gauge x 1
- Gravity 4P sensor wire (or Dupont wires) x 1
- **Software**
- Download and install the **DFRobot\_MAX17043 RaspberryPi library**
- RASPBIAN

## Connection Diagram



## Installation

1. Start the I2C interface of the Raspberry Pi. If it is already open, skip this step. Open Terminal, type the following command, and press Enter:

```
pi@raspberrypi:~ $ sudo raspi-config
```

Then use the up and down keys to select “5 Interfacing Options” -> “P5 I2C” and press Enter to confirm “YES”. Reboot the Raspberry Pi.

2. Installing Python libraries and git (networking required). If it is already installed, skip this step. In the Terminal, type the following commands, and press Enter:

```
pi@raspberrypi:~ $ sudo apt-get update
pi@raspberrypi:~ $ sudo apt-get install build-essential python-dev python-smbus git
```

3. Download the driver library and run it. In Terminal, type the following commands, and press Enter:

```
pi@raspberrypi:~ $ git clone https://github.com/DFRobot/DFRobot_MAX17043.git
pi@raspberrypi:~ $ cd ~/DFRobot_MAX17043/RaspberryPi/python
pi@raspberrypi:~/DFRobot_MAX17043/RaspberryPi/python $ python DFRobot_MAX17043.py
```

## Read Battery Voltage, Remaining Power and Set Low Power Interrupt Alert

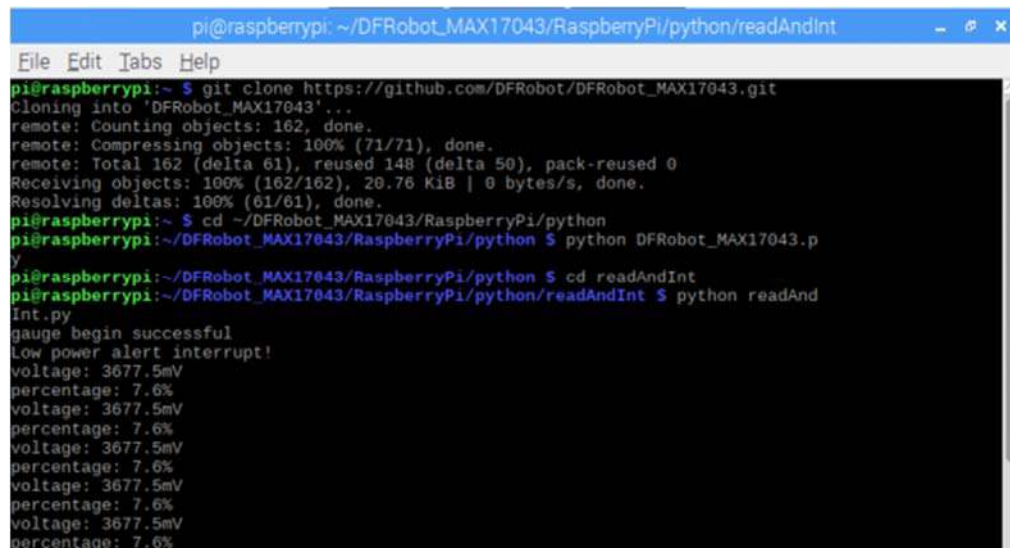
- Connect the module to the RaspberryPi according to the connection diagram. The battery can be connected to the 2P terminal or JST-PH2.0 2P connector. These two connectors are connected internally in parallel. The I2C address is fixed to 0x36.
- Install DFRobot\_MAX17043 RaspberryPi library.
- In the Terminal, type in the following commands and press Enter to run the sample code:

```
pi@raspberrypi:~/DFRobot_MAX17043/RaspberryPi/python $ cd readAndInt
pi@raspberrypi:~/DFRobot_MAX17043/RaspeberryPi/python/readAndInt $ python readAndInt.py
```



## Results

- RaspberryPi prints the current voltage (voltage), remaining power (percentage), and interrupt alert information (if any) every 2 seconds.
- Battery low power interrupt alert ALR instructions:
- The default value of the battery low power interrupt alert threshold is 32%. That is, when the remaining power is lower than 32%, a falling edge interrupt is generated on the ALR pin. This threshold can be set to any integer between 1-32 (corresponding to 1%-32%, respectively) with the function `setInterrupt()`.
- When the battery's initial remaining power is higher than interrupt alert threshold, the ALR pin is set high. If it falls below the threshold (due to discharge) , the ALR pin is pulled to low. The controller is triggered to print "Low power alert interrupt!", and then clear interrupt through `clearInterrupt()`, which causes ALR back to high immediately.
- When the battery's initial remaining power is below interrupt alert threshold, the ALR pin will generate an interrupt at the beginning.
- After the battery remaining power grows higher than interrupt alert threshold (due to discharge) , another interrupt will be generated when the power again falls below the threshold (due to discharge). If the `clearInterrupt()` is not called after the interrupt is occurred, the ALR pin will remain low regardless of the statue of the battery.



```
pi@raspberrypi: ~/DFRobot_MAX17043/RaspberryPi/python/readAndInt
File Edit Tabs Help
pi@raspberrypi:~$ git clone https://github.com/DFRobot/DFRobot_MAX17043.git
Cloning into 'DFRobot_MAX17043'...
remote: Counting objects: 162, done.
remote: Compressing objects: 100% (71/71), done.
remote: Total 162 (delta 61), reused 148 (delta 50), pack-reused 0
Receiving objects: 100% (162/162), 20.76 KiB | 0 bytes/s, done.
Resolving deltas: 100% (61/61), done.
pi@raspberrypi:~$ cd ~/DFRobot_MAX17043/RaspberryPi/python
pi@raspberrypi:~/DFRobot_MAX17043/RaspberryPi/python$ python DFRobot_MAX17043.py
pi@raspberrypi:~/DFRobot_MAX17043/RaspberryPi/python$ cd readAndInt
pi@raspberrypi:~/DFRobot_MAX17043/RaspberryPi/python/readAndInt$ python readAndInt.py
gauge begin successful
Low power alert interrupt!
voltage: 3677.5mV
percentage: 7.6%
voltage: 3677.5mV
percentage: 7.6%
voltage: 3677.5mV
percentage: 7.6%
voltage: 3677.5mV
percentage: 7.6%
voltage: 3677.5mV
percentage: 7.6%
voltage: 3677.5mV
percentage: 7.6%
```

## Application Examples

- Apply **Gravity: 3.7V Li Battery Fuel Gauge** to **Sunflower: Solar Power Manager 5V** to build a solar power management system. The battery fuel gauge can be used to monitor the charging of the battery with a solar panel in real time during the day, and the power consumption of the system during the night. Users can write their own code to record or print battery voltage and remaining power data, which can be employed to evaluate the "power balance" of the solar system in a day or even several months. These datum helps users to determine whether the nominal power of the panel or lithium battery power is large enough to fully support the entire system during the day and night.

