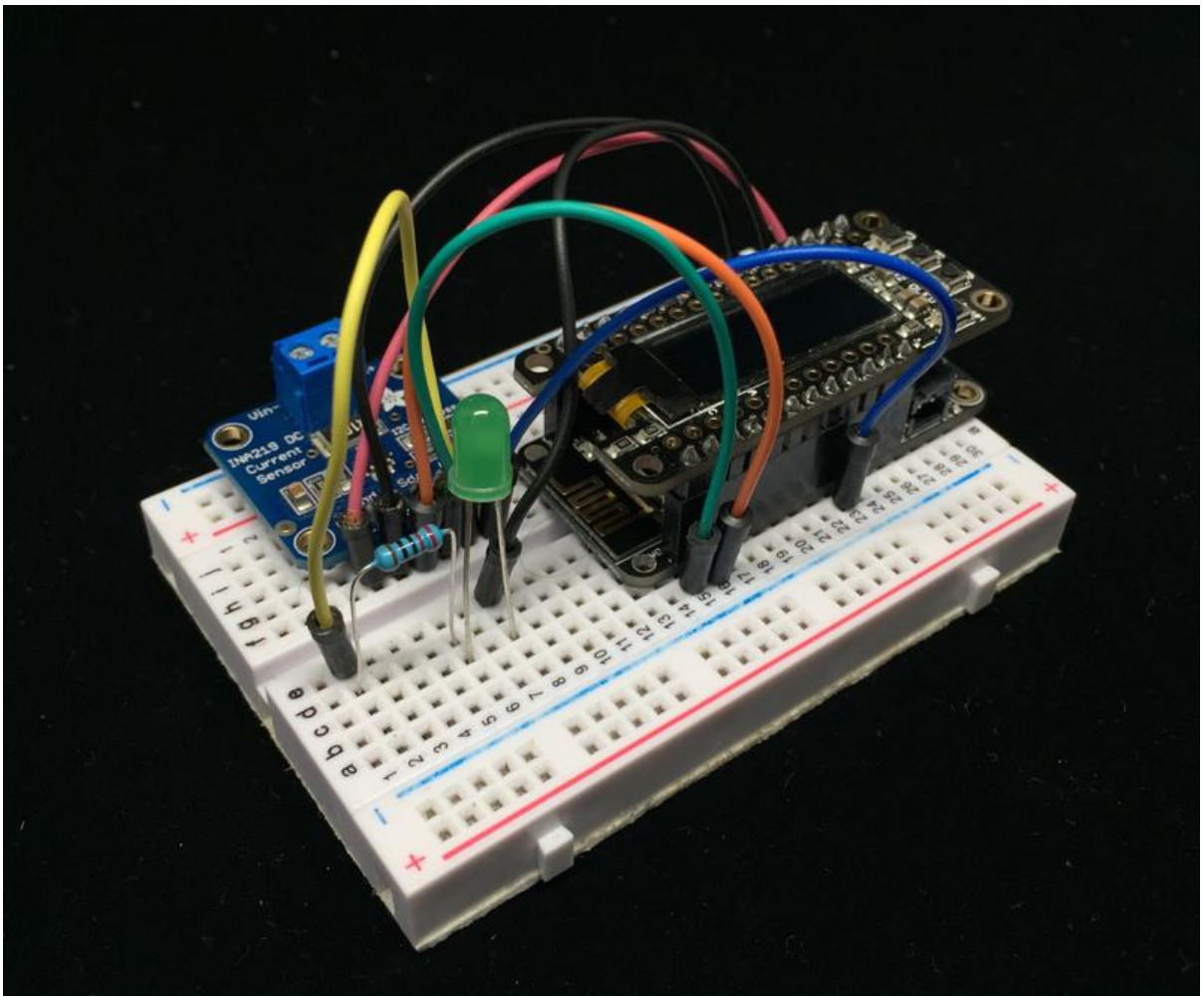
Build a Cloud-Connected ESP8266 Power Meter

Created by Marc-Olivier Schwartz



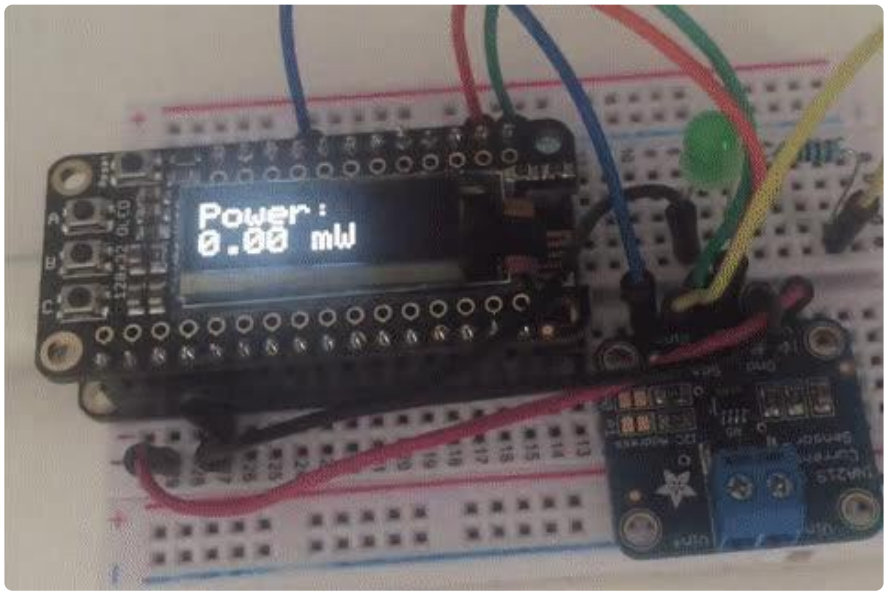
https://learn.adafruit.com/build-a-cloud-connected-esp8266-power-meter

Last updated on 2021-11-15 06:41:53 PM EST

Table of Contents

Introduction

Controlling the electrical consumption in your home is one of the most important thing you can do, both because of environmental concerns & to reduce the electricity bill at the end of the month. There are countless of electrical power meters out there, but in this guide, I'll show you how to build your own, and to use the ESP8266 feather board to measure how much power a single device is using. Note that this guide is about measuring power for DC (Direct Current) devices only.



Here, we are going to do something different: we are going to measure the power used by a device, and then display it right on top of the ESP8266 board, user the featherwing OLED add-on board. This way, you'll be able to build your own power meter based on the ESP8266, that is completely independent from any external components. As an additional function, we'll also send the data on Adafruit IO so it can be monitored online. Let's start!

Hardware Configuration

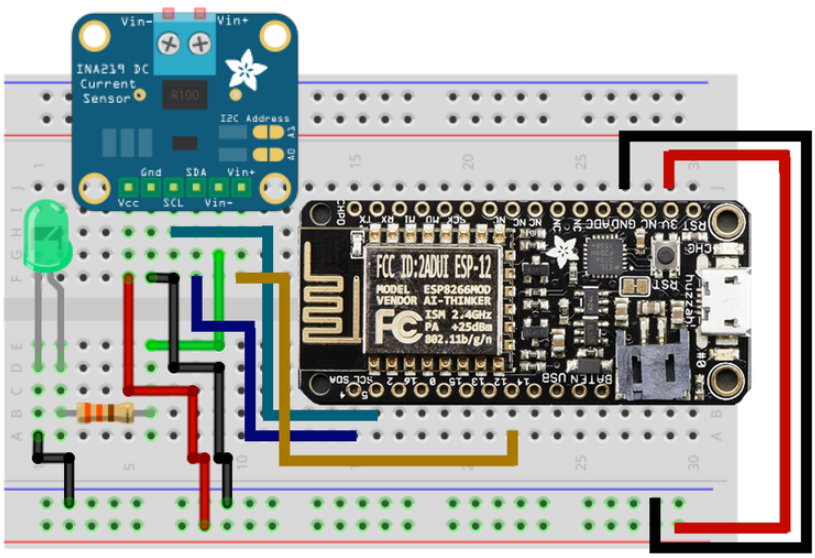
Let's first see what we need to build this project. As the center of the project, I used the Adafruit Feather HUZZAH ESP8266 board.

To measure the voltage & current of the measured device, I used an Adafruit breakout board for the INA219 sensor. To display the data, I used the Adafruit featherwing 128×32 OLED add-on, as it is very easy to use with the ESP8266 feather board.

As a test device, I will use a simple LED, along with a 330 Ohm resistor. But this could also be any DC device you are using in your home, for example a strip of LEDs.

Of course, you will need the usual breadboard & jumper wires to make the necessary connections.

Let's now see how to assemble the hardware for this project. Here is a schematic to help you out:

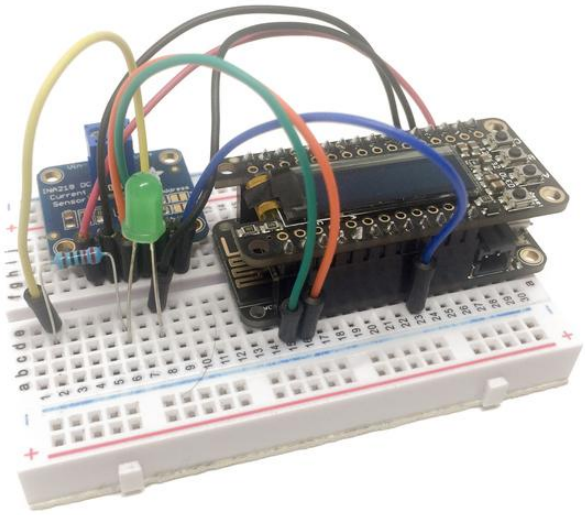


First, put the ESP8266 board on the breadboard, and mount the featherwing OLED add-on on it.

For the INA219 sensor, you first need to connect the power (VCC & GND) to the ESP8266 power. For convenience, I connected the ESP8266 VCC & GND pins to the two power rails of the breadboard. Then, you need to connect the SCL pin to the ESP8266 SCL pin, and the SDA pin to SDA pin on the ESP8266.

Finally, for the LED, we need to make the current of the LED 'flow' through the INA219 sensor, so it can be measured. To do so, first connect pin 12 of the ESP8266 board to the Vin+ pin of the sensor. Then, connect the Vin- pin of the sensor to the resistor, in series with the anode of the LED. Finally, connect the other pin of the LED to the ground.

This is the final result:



Configuring the Project

We are now going to configure the project we just assembled. On the software side, you will need the latest version of the Arduino IDE, that you can get from:

https://www.arduino.cc/en/Main/Software (https://adafru.it/fvm)

To be able to use the ESP8266, you will need to have the ESP8266 boards definitions installed. You can add those definitions to your Arduino IDE by following the instructions at:

https://github.com/esp8266/Arduino (https://adafru.it/eSH)

You can also learn more about configuring the Adafruit feather ESP8266 HUZZAH board at:

https://learn.adafruit.com/adafruit-feather-huzzah-esp8266/ (https://adafru.it/nEN)

You will also need the following Arduino libraries, that you can install from the Arduino library manager:

- Adafruit INA219
- Adafruit SSD1306
- Adafruit MQTT library
- Adafruit GFX library
- Adafruit BusIO

To learn how to install Arduino libraries, you can also check:

https://learn.adafruit.com/adafruit-all-about-arduino-libraries-install-use/ (https://adafru.it/nEO)

You will also need to have an Adafruit IO account. You can learn more about Adafruit IO & how to create an account at:

https://learn.adafruit.com/adafruit-io/ (https://adafru.it/nEP)

Let's now take care of the code for this project. As the code is quite long, I will only highlight the most important parts here, but you can find the complete code at:

https://github.com/openhomeautomation/power-meter-esp8266 (https://adafru.it/nEQ)

The code starts by importing the required libraries:

```
#include &lt;ESP8266WiFi.h&gt;
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"
#include &lt;SPI.h&gt;
#include &lt;Wire.h&gt;
#include &lt;Adafruit_GFX.h&gt;
#include &lt;Adafruit_SSD1306.h&gt;
#include &lt;Adafruit_INA219.h&gt;
```

Then, you need to set your WiFi network name & password:

```
const char* ssid     = "wifi-name";
const char* password = "wifi-pass";
```

You also need to set your Adafruit IO username and key:

```
#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883
#define AIO_USERNAME    "username"
#define AIO_KEY         "key"
```

Next, we define a feed called 'power' that will hold the measurements made by the board:

```
const char POWER_FEED[] PROGMEM = AIO_USERNAME "/feeds/power";
Adafruit_MQTT_Publish power = Adafruit_MQTT_Publish(&amp;mqtt, POWER_FEED);
```

Inside the loop() function, we continuously switch the LED on & off, and at the same time measure the power, send it to Adafruit IO, and display it on the OLED screen:

```
// LED ON
  digitalWrite(14, HIGH);

  // Measure
  current_mA = measureCurrent();
  power_mW = measurePower();

  // Publish data
  if (! power.publish(power_mW)) {
    Serial.println(F("Failed"));
  } else {
    Serial.println(F("OK!"));
  }

  // Display data
  displayData(current_mA, power_mW);
  delay(2000);
```

Note that here, we continuously switch the state of the LED, just to show the changes in the measured power on the OLED screen.

Of course, if you are using the project to measure the power flowing through a device not connected to the ESP8266, you will simply measure the power at regular intervals in the loop() function.

Here are the details of the function used to measure the power:

```
float measurePower() {

  // Measure
  float shuntvoltage = ina219.getShuntVoltage_mV();
  float busvoltage = ina219.getBusVoltage_V();
  float current_mA = ina219.getCurrent_mA();
  float 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("");

  // If negative, set to zero
  if (current_mA &lt; 0) {
    current_mA = 0.0;
  }

  return current_mA * loadvoltage;

}
```

And here are the details of the function used to display the data on the OLED screen:

```
void displayData(float current, float power) {

  // Clear
  display.clearDisplay();
  display.setTextSize(2);
```

```
  display.setTextColor(WHITE);

  // Power
  display.setCursor(0,0);
  display.println("Power: ");
  display.print(power);
  display.println(" mW");

  // Displays
  display.display();

}
```

Note that due to the small size of the screen, I am not displaying the current here, but you could also use this information instead of the power.

Testing the Project

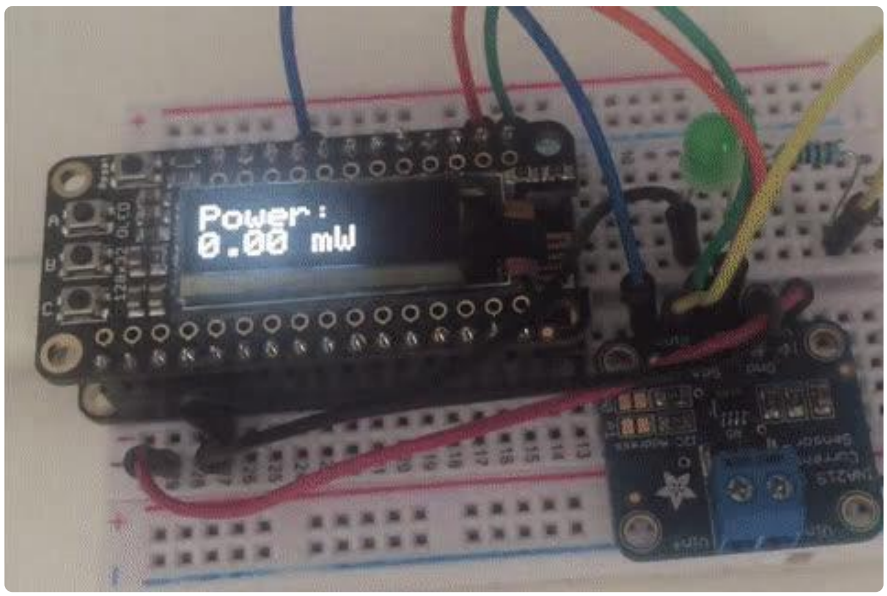
It's now time to finally test the project! You can of course grab the complete code from the GitHub repository of the project:

https://github.com/openhomeautomation/power-meter-esp8266 (https://adafru.it/nEQ)

Make sure to modify the code to include your own WiFi name and password, and also your Adafruit IO credentials.

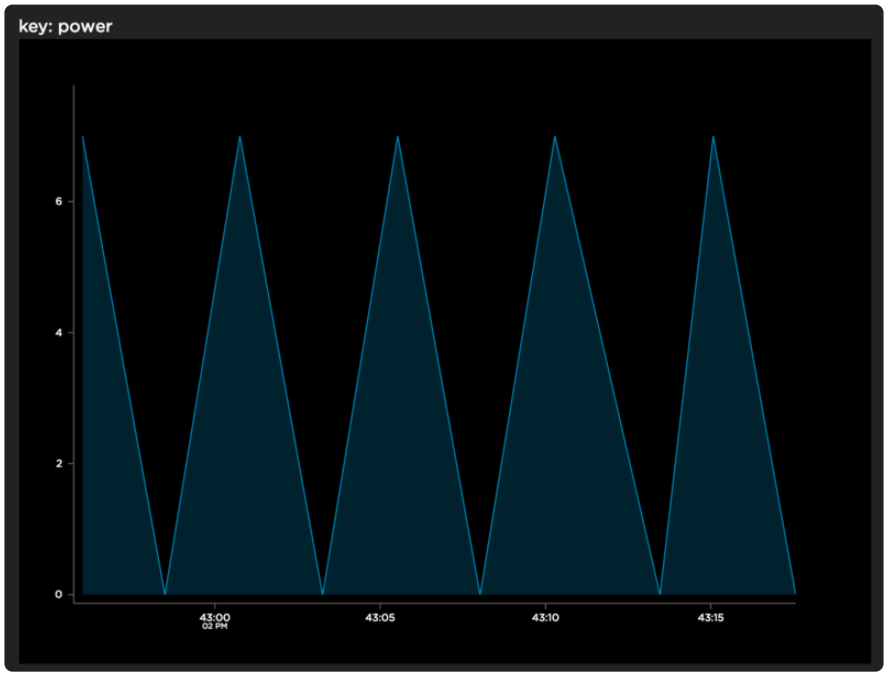
You can now upload the code to the board. Make sure that you selected the correct board (Adafruit HUZZAH ESP8266) inside the Arduino IDE.

You should observe the following result on the board:

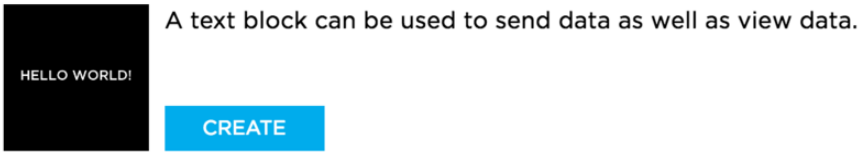


Now, remember that the project is also sending data to Adafruit IO! To visualise the incoming data sent by the board, log into Adafruit IO, and navigate to the 'feeds' menu. You should see a feed called 'power', as this is the name we set inside the code.

If you visualise the feed from Adafruit IO, it should like the following graph, as the LED is switching on & off continuously:



Before ending this guide, I will show you how to actually display the current power consumption of the device inside a dashboard. For that, open a dashboard on Adafruit IO or create a new one, and create a text block:



After that, link this block to the 'power' feed:

You should now see the instant consumption of the device connected to your project. For example, you should see 0.00 when the LED is off:

This is the same text block, showing the power consumption when the LED is on:

How to Go Further

In this guide, you learned how to build a power meter based on the ESP8266, using a featherwing OLED add-on to display the measured power. The project was also sending data directly to Adafruit IO.

You can now of course take what you learned in this article and use it in several other projects. You can for example use this project on several devices in your home, and have them all display the current power used by the device, while monitoring all the devices from an Adafruit IO dashboard.