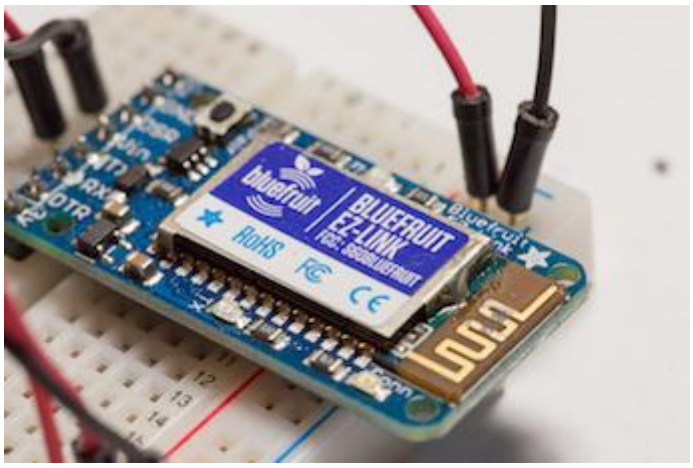
Bluetooth Temperature & Humidity Sensor

Created by Marc-Olivier Schwartz

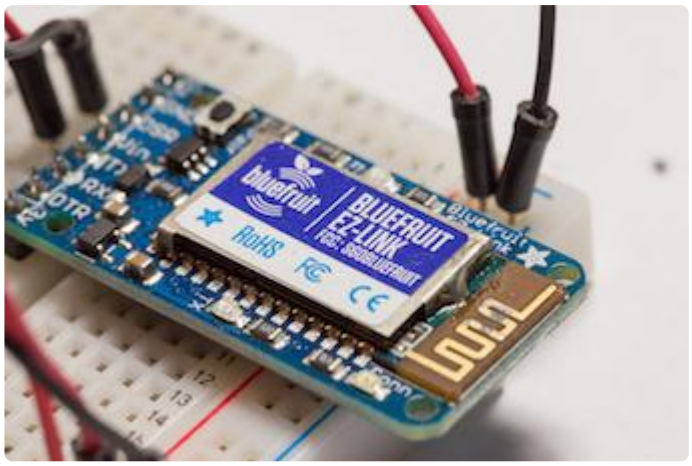


https://learn.adafruit.com/bluetooth-temperature-and-humidity-sensor

Last updated on 2023-08-29 02:29:04 PM EDT

Table of Contents

Introduction



Bluetooth devices are widely used in many consumers products, its the most popular wireless protocol for small point-to-point networking. Every laptop and just about every computer has Bluetooth classic built into it, so you often don't need a data receiver for a computer. And recently, the Bluefruit product family has made it even easier to integrate Bluetooth in an Arduino project. So why not use this technology in a simple home automation project?

Bluetooth is fast, low-power, and you can communicate with Bluetooth devices directly from a computer because they usually have built-in Bluetooth capabilities. The other nice thing is that with this project, you will be able to change the sketch running on your Arduino via the Bluetooth connection, without having to plug any cables!

In this project, you will learn how to connect a Bluetooth module to Arduino, transmit measurements from a temperature & humidity sensor to your computer, and display the data in a nice Python interface. Let's start!

Connections

The hardware configuration consists in two parts: connecting the Bluetooth EZ-Link module to the Arduino board, and then connecting the DHT22 temperature/humidity sensor.

The EZ-Link will take care of the wireless data transmission, and the DHT22 is a sensor. The Arduino sits between the two, reading data from the DHT22 sensor and then sending the data to the EZ-Link module.

Before you start this section we suggest going thru the tutorials we have for both these components!

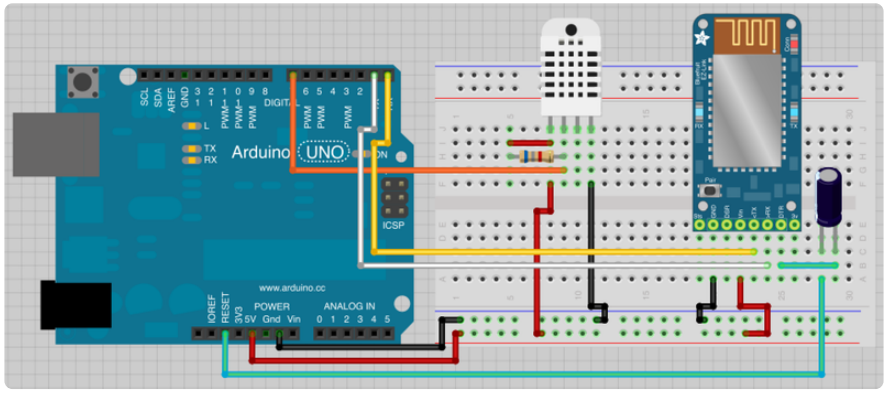
- EZ-Link tutorial ()
- DHT22 tutorial ()

Start by plugging the Bluetooth module on the breadboard. You then need to connect the power for the Bluetooth module: the 5V coming from the Arduino board, and the Ground pin.

Now, we have to make the connections from the Bluetooth board to the Arduino so that both can communicate via the Serial connection. Connect the Bluetooth TX pin to the Arduino RX (pin 0), and the RX pin to the Arduino TX (pin 1). You also have to connect the DTR pin of the Bluetooth board to the Arduino Reset pin (which is next to the 3.3V pin), via a 1uF capacitor (the negative side of the capacitor is marked with a gray line, and should be on the side of the Arduino Reset pin).

Finally, you need to plug the DHT temperature sensor to your project. Pin number 1 goes to the Arduino 5V, pin number 2 to Arduino pin 7, and pin number 4 to Arduino Ground. Finally, place a resistor (between 4.7K and 10K) between pin number 1 and 2 of the DHT sensor.

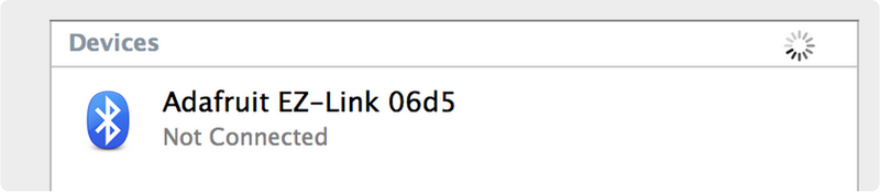
The following picture summarizes the connections for this project:



Arduino sketch

For this project, you will need the Arduino IDE () installed, along with the DHT sensor library () and the Python PySerial module ().

We'll first test the Bluetooth module to see if everything is connected correctly. You need to pair the Bluetooth module with your computer first. It depends on your OS, but you will usually have a "Bluetooth preferences" menu to search for new Bluetooth devices:



Once the device is paired with your computer, you can reopen the Arduino IDE and test if the Bluetooth connection is working. In Tools>Serial Port, you should have new choices for your Bluetooth device. Choose the second one:

You can now work with the Arduino IDE as if the Arduino board was directly connected to your computer: you can plug your Arduino board to an external source of power like a battery, and use the Bluetooth connection to upload sketches.

To try it out, just load the "Blink" sketch, and click on upload: after a while the sketch should be uploaded (it takes longer than with a USB cable) and the onboard LED of the Arduino Uno should blink.

We now need to write the code for the Arduino, so it measures the temperature & humidity when it receives a given command on the Serial port. This command will later be sent by your computer, but for now we'll just make a simple test to make sure the Arduino part is working.

The core of the sketch is to make the Arduino answer with the temperature & humidity measurement on the Serial port when a given character is received. I chose the character "m" for "measurement" to make the Arduino send the measurements over the Serial port. This is the part of the code that does exactly that:

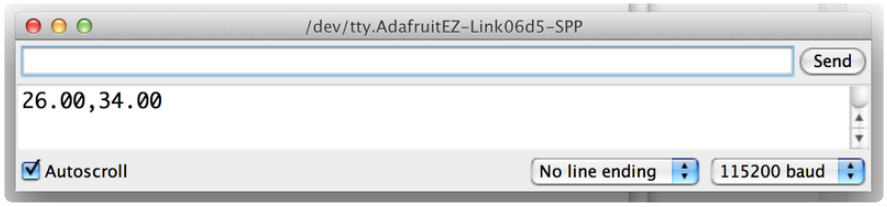
```
byte c = Serial.read ();

// If a measurement is required, measure data and send it back
if (c == 'm'){

   int h = (int)dht.readHumidity();
   int t = (int)dht.readTemperature();

   // Send data (temperature,humidity)
   Serial.println(String(t) + "," + String(h));
}
```

This is the complete sketch for this part:

```
// Bluetooth temperature sensor
#include "DHT.h"
```

```
// Pin for the DHT sensor
#define DHTPIN 7
#define DHTTYPE DHT22
// #define DHTTYPE DHT11

// Create instance for the DHT sensor
DHT dht(DHTPIN, DHTTYPE);

// Setup
void setup(void)
{
  dht.begin();
  Serial.begin(115200);
}

void loop(void)
{
    // Get command
    if (Serial.available()) {

      // Read command
      byte c = Serial.read ();

      // If a measurement is requested, measure data and send it back
      if (c == 'm'){

          int h = (int)dht.readHumidity();
          int t = (int)dht.readTemperature();

          // Send data (temperature,humidity)
          Serial.println(String(t) + "," + String(h));
      }
  }
}
```

Now upload the sketch (via Bluetooth of course!), open the serial monitor, and type in "m" and click send. This is what you should see on the Serial port:



This means whenever the Arduino will receive the character "m", it is going to return to correct measurements (in this case the temperature was at 26 degrees Celsius and the humidity was at 34 %).

Python interface

Finally, we need to build an application running on your computer to send the order to get new measurements from the Arduino board, retrieve the data, and display it on your screen. I chose Python for this interface but it's quite easy to interface with the Serial port with PySerial, and it is also easy to build an interface with Tkinter which is installed by default with Python.

I won't detail every piece of the code here because it's way too long, but you can of course find the complete code in the GitHub repository for this project ().

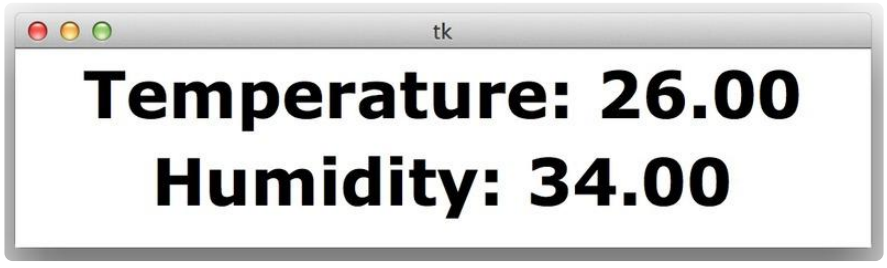
The Python code starts by initialising the Serial connection:

```
serial_speed = 115200
serial_port = '/dev/cu.AdafruitEZ-Link06d5-SPP'
ser = serial.Serial(serial_port, serial_speed, timeout=1)
```

Then, the core of the Python script is the measure() function that is continuously executed every second:

```
# Measure data from the sensor
def measure(self):

   # Request data and read the answer
   ser.write("m")
   data = ser.readline()

   # If the answer is not empty, process &amp;amp; display data
   if (data != ""):
      processed_data = data.split(",")
      self.temp_data.set("Temperature: " + str(processed_data[0]))
            self.temperature.pack()

      self.hum_data.set("Humidity: " + str(processed_data[1]))
            self.humidity.pack()

   # Wait 1 second between each measurement
   self.after(1000,self.measure)
```

Then, the rest of the script simply displays this data on a simple Tkinter window. You can get the complete code from the GitHub repository for this project (). Finally, type "python sensor_gui.py" in a terminal. This is what you should get:



Congratulations, you just built a Bluetooth temperature & humidity sensor ! Of course, you can use the code found in this project to interface other sensors to your computer via Bluetooth, for example ambient light sensors or motion sensors.

Each new Bluetooth module that you add will appear as a new Serial Port on your computer, so you can perfectly have several of these Bluetooth-based sensors in your home and build a whole home automation system from it!