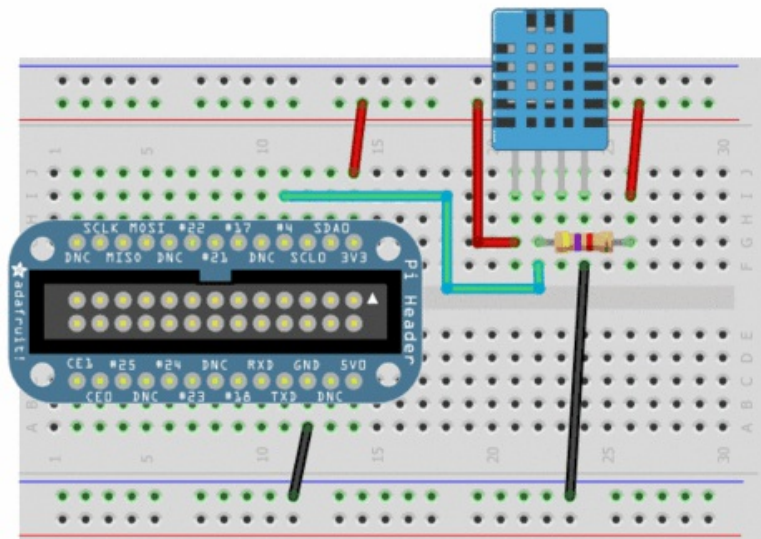


DHT Humidity Sensing on Raspberry Pi with GDocs Logging

Created by Ladyada



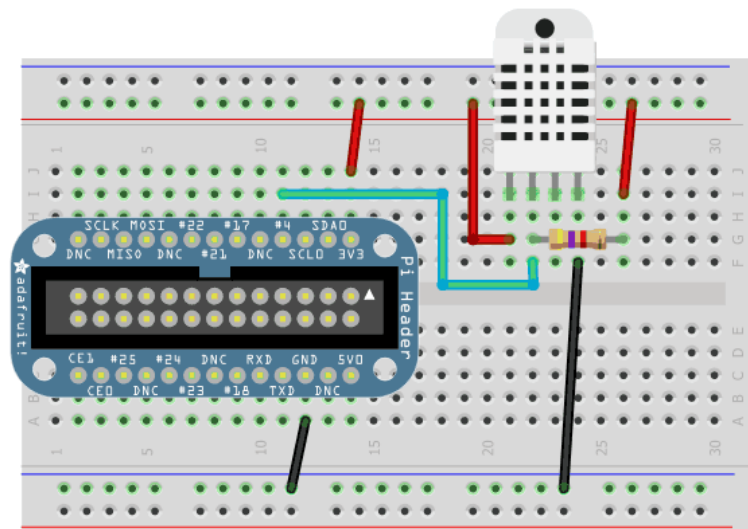
Last updated on 2013-09-02 12:30:26 PM EDT

Guide Contents

Guide Contents	2
Overview	3
Wiring	4
Wiring up DHT humidity sensors	4
Software Install	6
Downloading the Code from Github	6
Testing the C Code	6
Adapting the DHT C code	7
Connecting to Google Docs	8
Create and prepare spreadsheet	8
Run Python code on Pi	8

Overview

This tutorial is a first attempt to develop a DHT interface driver. It is not guaranteed to work, and is for experimentation and hacking!



Time to start exploring more sensors with the Raspberry Pi! Today we'll be checking out the [DHT11](http://adafru.it/386), [DHT22](http://adafru.it/385) and [AM2302](http://adafru.it/393) humidity and temperature sensors available from Adafruit

In this tutorial we'll be showing how to utilize C for high-speed GPIO polling to handle bit-banged sensor output. Many low cost sensors have unusual output formats, and in this case, a "Manchester-esque" output that is not SPI, I2C or 1-Wire compatible must be polled continuously by the Pi to decode. Luckily, the C GPIO libraries are fast enough to decode the output.

Once we have that working, we add the fun of Python to update a google spreadsheet live with the temperature/humidity data. This project would be the great basis for home or garden automation!

You can check out our spreadsheet here, it wont be updated live after Aug 24 2012 but it will show you the format of data you get (<http://adafru.it/aOU>)

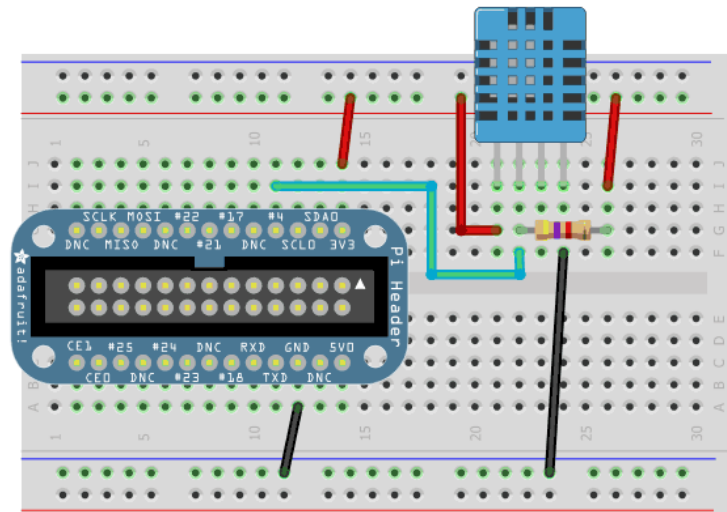
Wiring

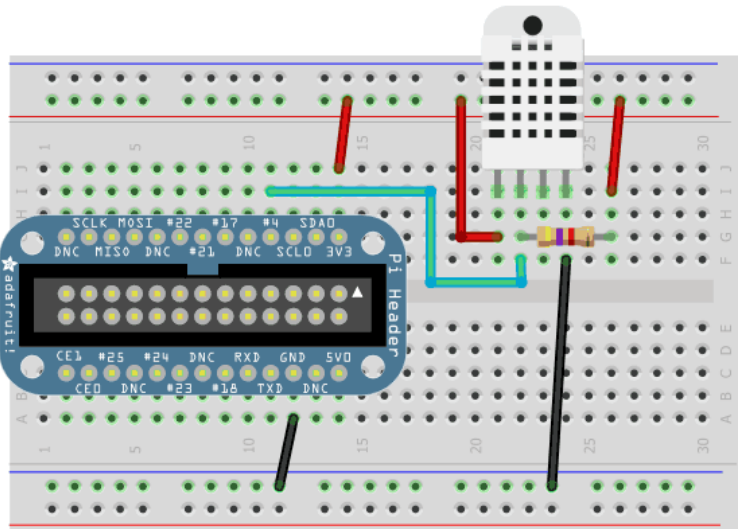
Wiring up DHT humidity sensors

Its easy to connect these sensors to your Raspberry Pi. Our code can use any GPIO pin, but we'll be using GPIO #4 for our diagrams and code. Once you have it working, you can simply adapt the code to change to any other GPIO pin (e.g. pin #18). You can also have as many DHT sensors as you want **but** they cannot share the data pin - each sensor needs a unique data pin!

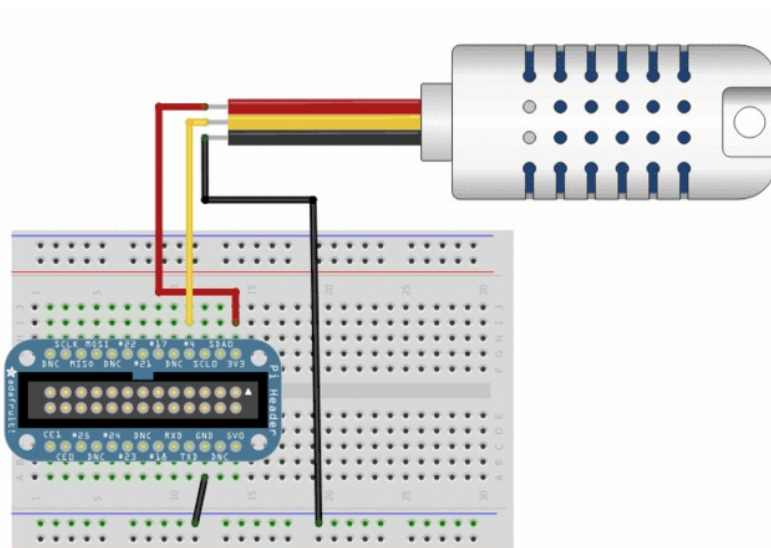
For DHT11 and DHT22 sensors, don't forget to connect a 4.7K - 10K resistor from the data pin to VCC

& if 4.7K doesnt work, try 10K





If your sensor has a white wire, leave it disconnected



Software Install

This tutorial is a first attempt to develop a DHT interface driver. It is not guaranteed to work, and is for experimentation and hacking!

The Python and C code to work with Adafruit's LED Backpacks on the Pi is available on Github at <https://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code> (<http://adafru.it/aOg>)

We use some C code to talk to the DHT sensors since they require extremely fast timing to read, and then Python code to update the Google Doc with our data!

Downloading the Code from Github

The easiest way to get the code onto your Pi is to hook up an Ethernet cable, and clone it directly using 'git', which is installed by default on most distros. Simply run the following commands from an appropriate location (ex. "/home/pi"):

```
$ git clone git://github.com/adafruit/Adafruit-Raspberry-Pi-Python-Code.git
$ cd Adafruit-Raspberry-Pi-Python-Code
$ cd Adafruit_DHT_Driver
```

Testing the C Code

First up, we'll test the C code to verify it works. Wire up a DHT sensor connected up so the data line is on GPIO #4, and powered to 3.3V and ground. If you are using a DHT11 or DHT22 you should add a 4.7K - 10KΩ resistor between the Data pin and the VCC pin.

Then run

```
sudo ./Adafruit_DHT type pin#
```

For example, we'll have an AM2302 connected to GPIO #4 so we'll run

```
sudo ./Adafruit_DHT 2302 4
```

You can change the GPIO pin on the command line later to any Pi pin if you wish, but we suggest starting with #4

You may have to wait a few seconds for the sensor to 'boot up' and then try a few times before it will respond with a temperature and humidity

```
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver
File Edit View Options Transfer Script Tools Window Help
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver # sudo ./Adafruit_DHT
usage: ./Adafruit_DHT [11|22|2302] GPIOpin#
example: ./Adafruit_DHT 2302 4 - Read From an AM2302 connected to GPIO #4
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver # sudo ./Adafruit_DHT 2302 4
Using pin #4
Data (40): 0x1 0xaa 0x1 0x2 0xae
Temp = 25,6 °C, Hum = 42,6 %
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver #
```

The DHT series of sensors will only respond every 2 seconds so if you are not getting data, be sure to wait 3 seconds before trying again!

```
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver
File Edit View Options Transfer Script Tools Window Help
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver # sudo ./Adafruit_DHT 2302 4
Using pin #4
Data (0): 0x0 0x0 0x0 0x0 0x0
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver # sudo ./Adafruit_DHT 2302 4
Using pin #4
Data (40): 0x1 0xa4 0x1 0x3 0xa9
Temp = 25,6 °C, Hum = 42,0 %
pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver #
```

Adapting the DHT C code

Since the DHT sensors use a low-level "Manchester-esque" timing protocol to send data, we need to be able to read the pin they are connected to at very high speeds. The python libraries aren't fast enough, but the low level C libraries are! The code we wrote is a good example of how to deal with bitbang in user space, without the need to write a kernel driver

Grab the lowlevel BCM2835 C Library from

<http://www.open.com.au/mikem/bcm2835/index.html> (<http://adafru.it/aOT>)

```
$ wget http://www.open.com.au/mikem/bcm2835/bcm2835-1.8.tar.gz
$ tar -zxvf bcm2835-1.8.tar.gz
$ cd bcm2835-1.8
$ ./configure
$ make
$ sudo make install
```

Then compile the Adafruit_DHT program with

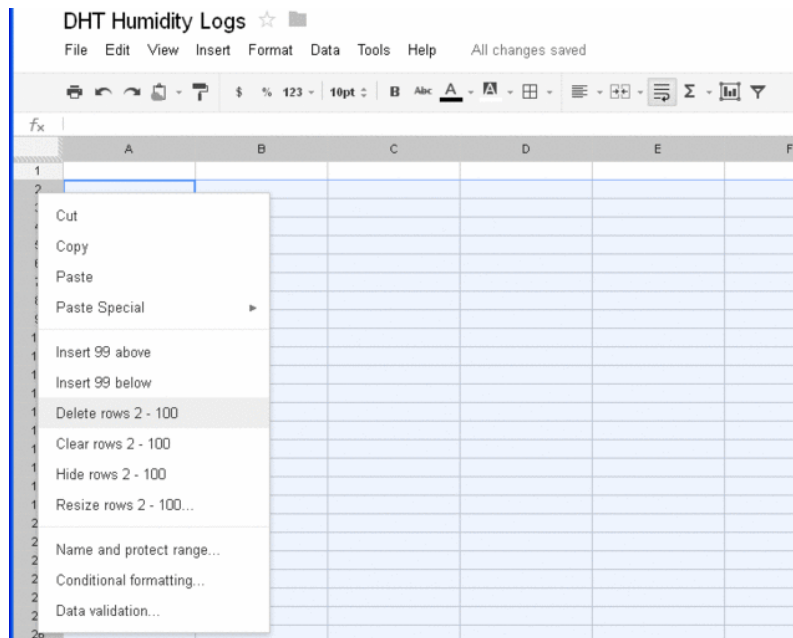
```
gcc Adafruit_DHT.c -I bcm2835 -std=gnu99 -o Adafruit_DHT
```

Connecting to Google Docs

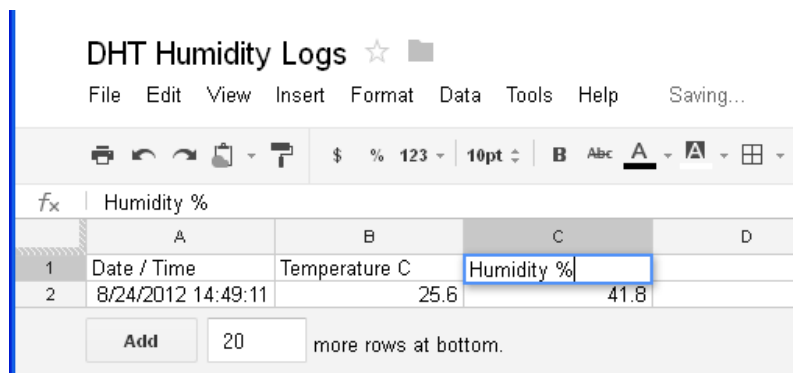
Create and prepare spreadsheet

First up you will need to sign up for Google Docs and create a spreadsheet. We're going to call ours DHT Humidity Logs.

Once you've created it, delete all but one line (since we don't want 99 empty rows)



Then make the one remaining line a header with row names



Run Python code on Pi

First up we will have to install the **gsread** python library, which will do the heavy lifting of

connecting to google docs and updating the spreadsheet! With your Pi connected and online, run the following:

```
wget http://pypi.python.org/packages/source/g/gspread/gspread-0.1.0.tar.gz
tar -zxvf gspread-0.1.0.tar.gz
cd gspread-0.1.0
sudo python setup.py install
```

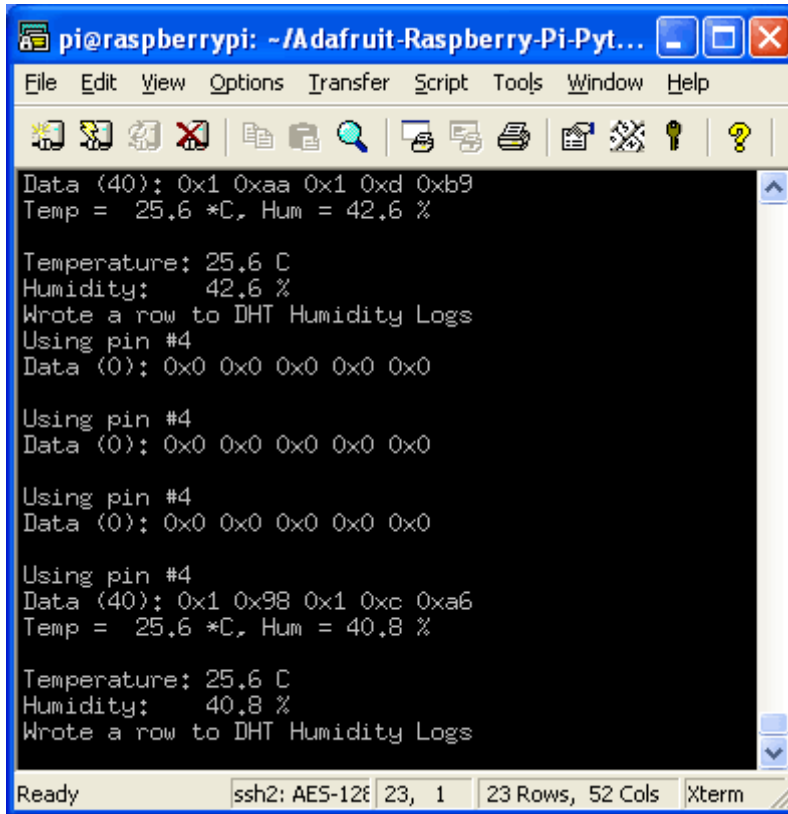
Next, in the **Adafruit-Raspberry-Pi-Python-Code/Adafruit_DHT_Driver** directory again, edit **Adafruit_DHT_googledocs.ex.py**

```
# =====
# Google Account Details
# =====

# Account details for google docs
email    = 'you@somewhere.com'
password = '$hhh!'
spreadsheet = 'SpreadsheetName'
```

You'll have to put in your google email account, password and the spreadsheet name. You can also use the spreadsheet key if you don't want to refer to it by name, look lower down for the **worksheet = gc.open_by_key()** line and uncomment it, changing the key

That's pretty much it! Just run the python script with **sudo**. Whenever there's a hiccup with the sensor, it will wait 3 seconds and then try again, it doesn't fail that often anyways. You can edit the update delay at the last line of the script by editing **time.sleep(30)** to a different # of seconds



The image shows a terminal window titled "pi@raspberrypi: ~/Adafruit-Raspberry-Pi-Pyt...". The window contains the following text:

```
Data (40): 0x1 0xaa 0x1 0xd 0xb9
Temp = 25,6 *C, Hum = 42,6 %

Temperature: 25,6 C
Humidity: 42,6 %
Wrote a row to DHT Humidity Logs
Using pin #4
Data (0): 0x0 0x0 0x0 0x0 0x0

Using pin #4
Data (0): 0x0 0x0 0x0 0x0 0x0

Using pin #4
Data (0): 0x0 0x0 0x0 0x0 0x0

Using pin #4
Data (40): 0x1 0x98 0x1 0xc 0xa6
Temp = 25,6 *C, Hum = 40,8 %

Temperature: 25,6 C
Humidity: 40,8 %
Wrote a row to DHT Humidity Logs
```

The terminal status bar at the bottom shows "Ready", "ssh2: AES-128", "23, 1", "23 Rows, 52 Cols", and "Xterm".

You can open the spreadsheet on your computer and watch it update live!

You can also see our spreadsheet [here](http://adafru.it/aOU), it wont be running live after Aug 24, 2012 but it gives you an idea of the data format (<http://adafru.it/aOU>)