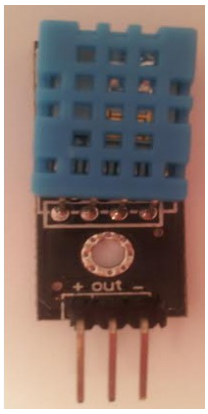
DHT11 HUMIDITY & TEMPERATURE SENSOR MODULE WITH A13-SOM-LTE BOARD

THE MODULE

The module used here is composed of the DHT11 sensor together with a 5K pull-up resistor and other required components. The sensor itself includes a resistive humidity measurement component, a 8-bit micro-controller and an NTC temperature measurement component. It is calibrated to deliver a digital signal which is available on the out wire of the 3-wire module :



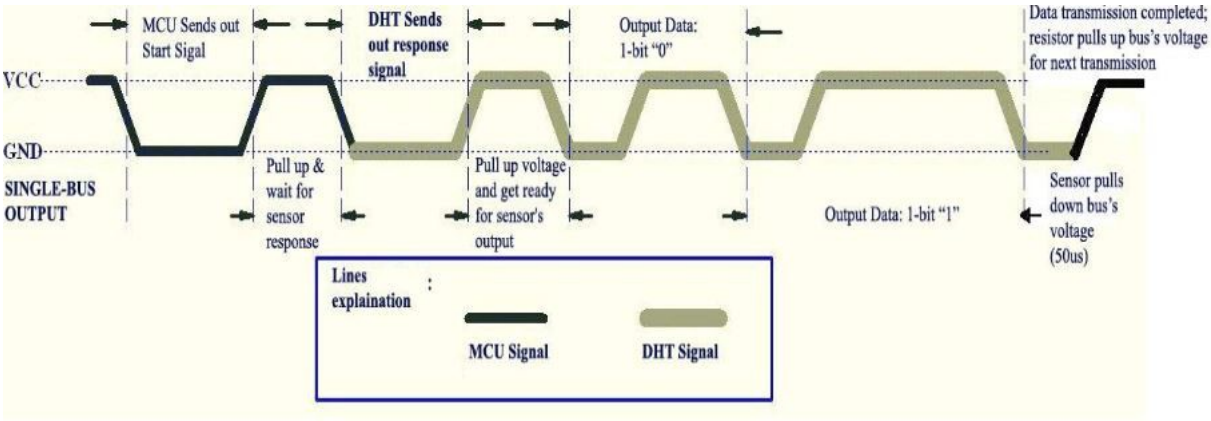
DHT11 is an easy to handle and cheaper module (compared to its big brother (DHT11)) but unfortunately with less resolution (+-1% for humidity and +-1 °C for temperature). Here are some technical specifications of the sensor used here, more exhaustive specification can be found in this document :

Power Supply□	3.3~5.5V
DCOutput□	4 pin single row\J
Measurement Range□	Humidity 20-90%RH□ Temperature 0~50JI \J
Accuracy□	Humidity +-5%RH□ Temperature +-2JI \J
Resolution□	Humidity 1%RH□ Temperature 1JI
Interchangeability□	Fully Interchangeable
Long-Term Stability□	<±1%RH/year

USING DHT11 WITH 4GKIT

To use the DHT11 module we should first understand the communication protocol it uses to exchange data with its master (the MCU here).

As shown in the following figure, the default status of the out pin is high. To receive measurements from the sensor the MTU should send a “start signal” by pulling down the out pin for at least 18 ms then pulling it up for 20 microseconds.

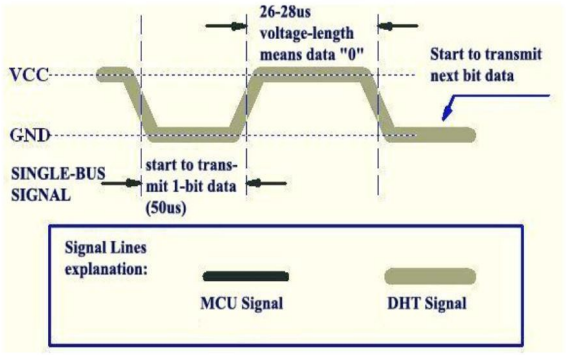


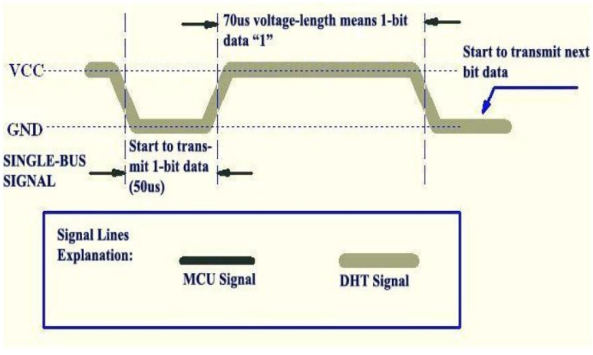
Right after the sensor sends a response signal and then the measurements data which is structured as follows :

```
0                   1                   2                   3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Humidity   |        0      | Temperature   |        0      |   Checksum    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Each of the three main fields is eight bits long. The zero bytes are the decimal bytes, as the resolution is the unit then the it is put to zero.

One thing to have in mind while processing the data from DHT11 is the coding of a data bit. In fact, every bit begins with a 50us low-voltage-level signal followed by a high-voltage-level signal which duration depends on the encoded bit. If it is equal to 0 then the signal lasts 26 to 28us else it lasts 70us. This is gonna be translated to a different number of ones between zero and one bits (greater for bits equal to 1) depending on the sampling frequency of the master.





Enough talking about the DHT11 module, let us see how to use it with the 4GKit.

But is your 4GKit ready to use GPIOs?

Normally yes, you can access the state of any GPIO pin via the SysFS interface by reading file value in the /sys/class/gpio/<name of GPIO pin>/ directory (as long as you the configuration within the script.bin file allows you to do so for the specific pin, see this page for more details, to edit the configuration on the 4Gkit board you can use the script chscr.sh as shown in this HowTo).

Any programming / scripting language that handles files would allow you to use GPIOs. In particular, OLIMEX offers a Python library that makes using GPIOs even easier. Hence we chose to use a Python script so as to communicate with the DHT11 module.

To get Python's GPIO Library ready for use open a command line on your 4GKit and type the following

commands :

```
# Download the libray's file
wget https://pypi.python.org/packages/source/p/pyA13/pyA13-0.1.12.tar.gz

# Unzip it
tar -xvzf pyA13-0.1.12.tar.gz

# Get additional necessary python header files
sudo apt-get install python-dev

# Install the GPIO Library
cd pyA13-0.1.12
sudo python setup.py install
```

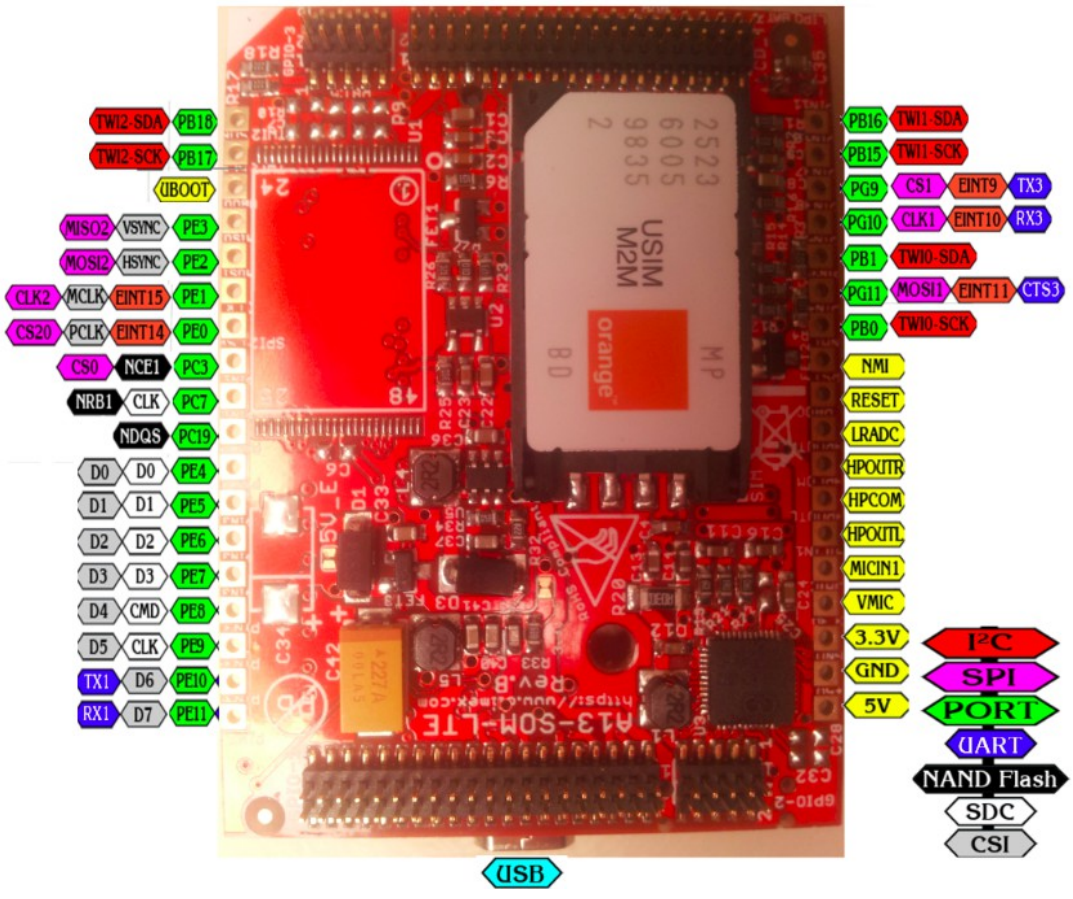
To check everything worked fine try importing GPIO package in python as follow :

```
python
>>> from pyA13.gpio import gpio
```

If no error is shown you can cheer up as your 4GKit is ready to allow you play with GPIOs.

One last step, you need to wire your sensor if you want it to work :)

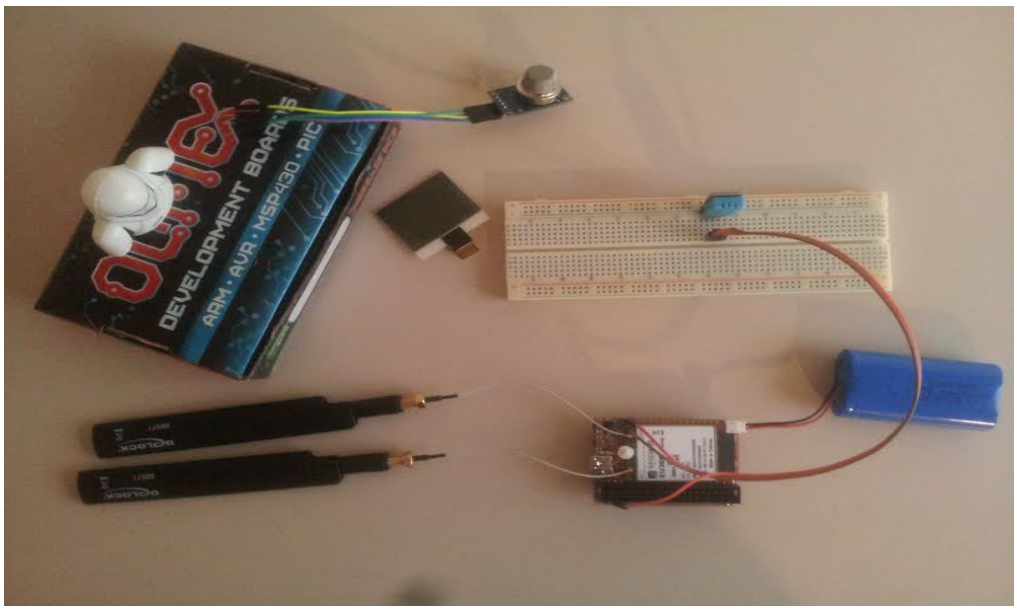
Here is a detailed Map of the different inputs on the daughter board of the 4GKit :



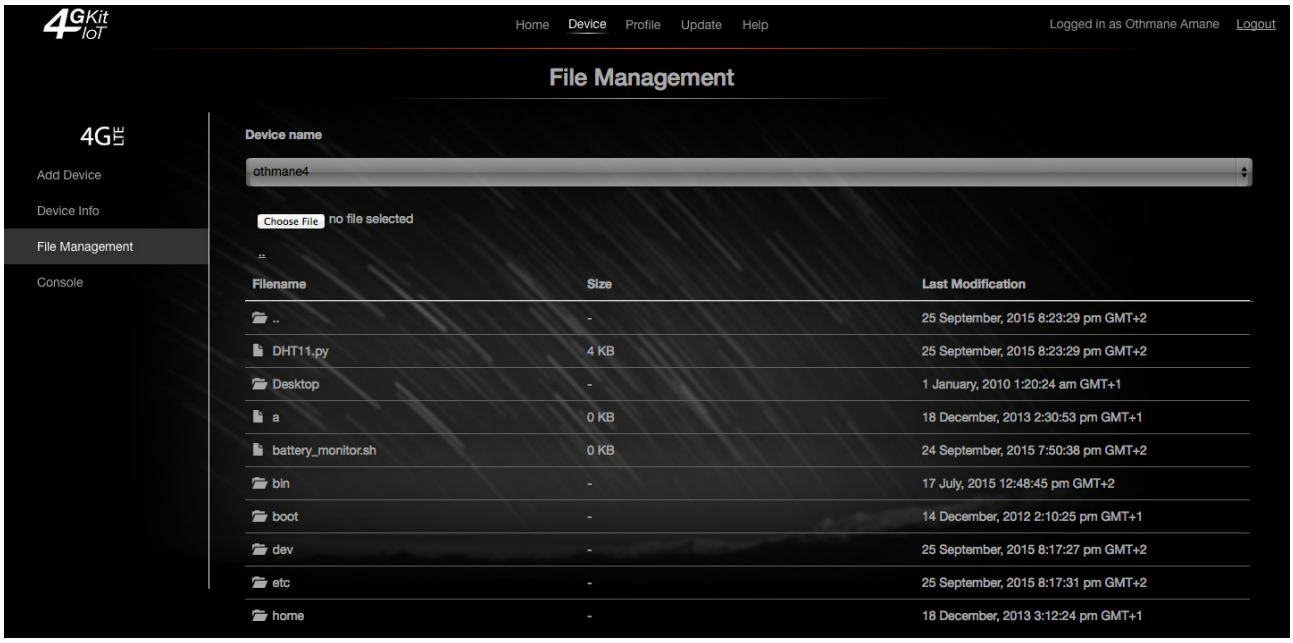
And the wiring for our DHT11 sensor :

4GKit		DHT11 Module\l
3.3v	———	+ (VCC)
GND	———	- (GND)
GPIO	———	OUT (DATA)

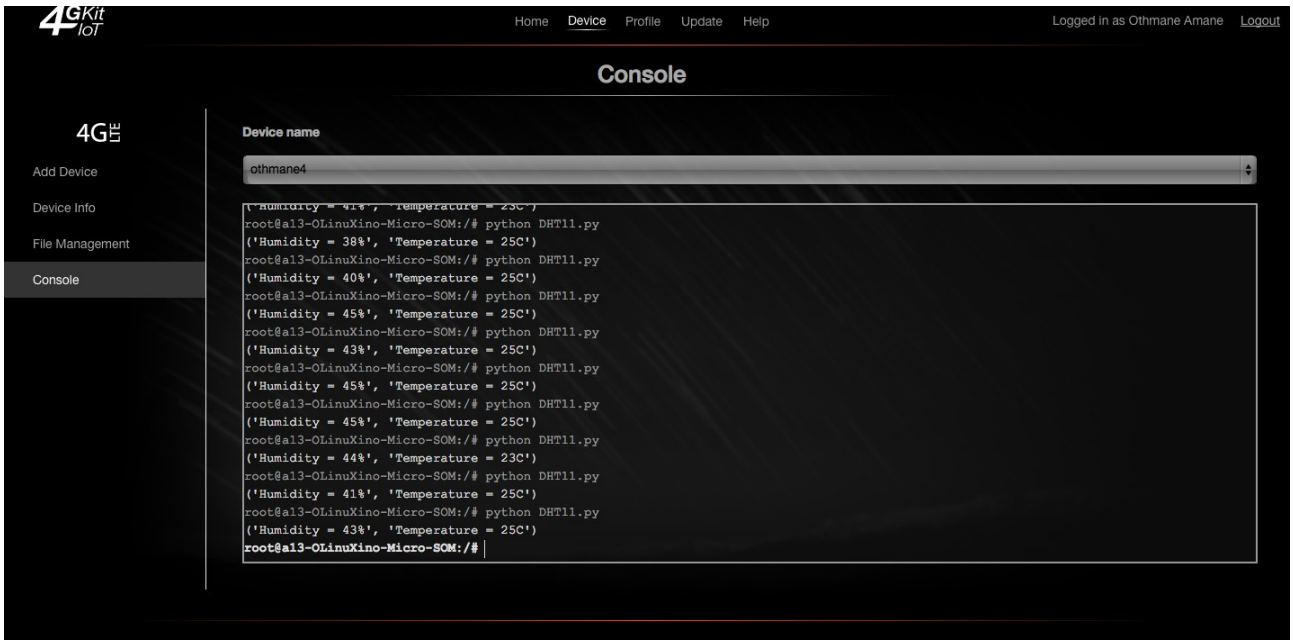
And a picture of the whole setup :



Time to code! Or you can just use the ready for use Python script. You only need to drop it to your space on 4gkit.orange.com



then you launch it and you get both temperature and humidity (sometimes an error my occur because the board may lose data as it is not running at real time but that should not happen often)



Enjoy using the 4GKit!