Jawahar Education Society's

A.C. Patil College of Engineering, Kharghar

Department of CSE (IoT-CS BC)

Subject: Edge and Fog Computing

Experiment No. 3

Aim Cloud based Temperature and Humidity Monitoring Application

OBJECTIVE :

a. To understand the working of IoT cloud platform
b. To develop comprehensive approach towards building small low cost IoT application.

S/W PACKAGES AND H/W USED:

Raspbian OS, Raspberry Pi 3, DHT11

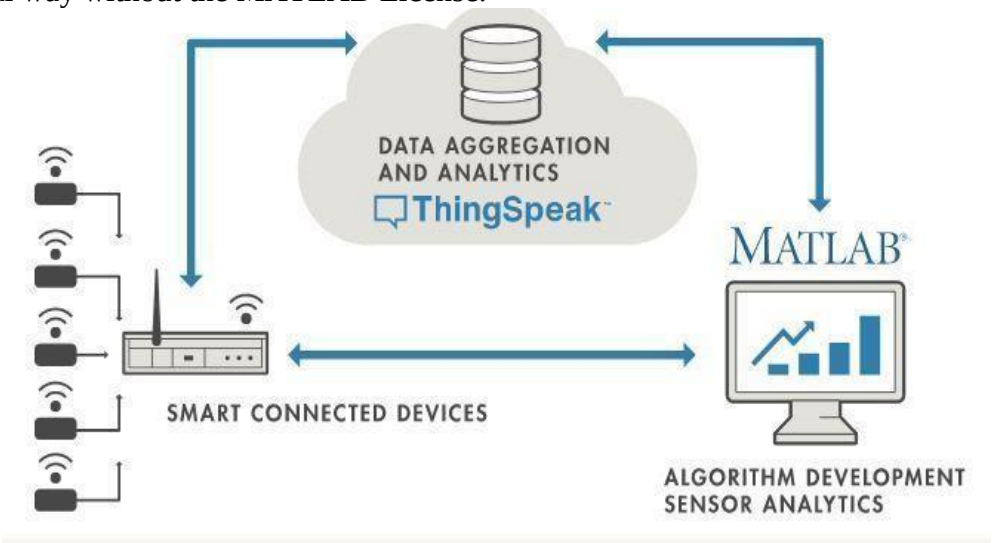
THEORY:

1. ThingSpeak

ThingSpeak is a IoT cloud platform which uses simple HTTP Protocol to transfer, store and retrieve information from different sensors. Also, the ThingSpeak Application allows us to log the sensor data, track locations and even social networking of things.

Another important thing (or rather a unique feature) about ThingSpeak is its support from MATLAB. The close relationship between ThingSpeak and MATLAB has lead to integrate several key features of MATLAB into the ThingSpeak Application.

One such feature is to analyse and visualize the user data i.e. the sensor data in a graphical way without the MATLAB License.

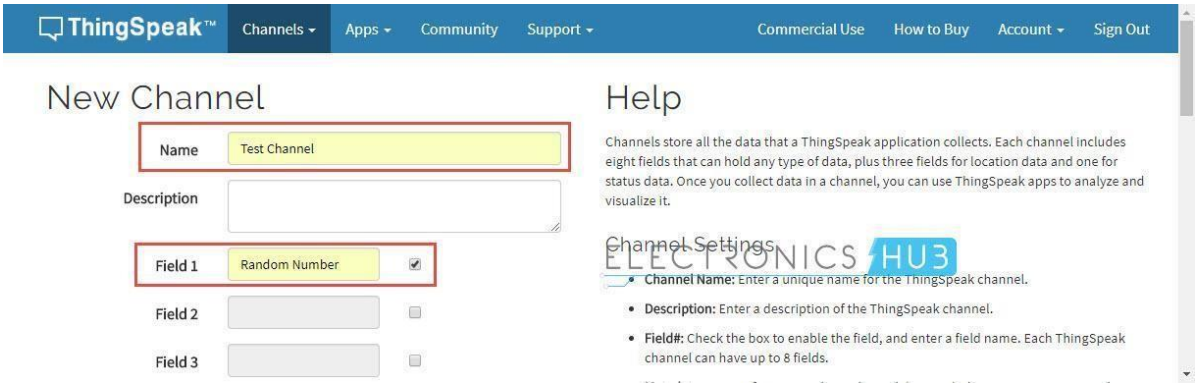


1.1 Creating ThingSpeak Account

- Create an account with ThingSpeak (www.thingspeak.com). Since the collaboration with MATLAB, you can use your MathWorks credentials to login to ThingSpeak. (NOTE: The MathWorks Account can be used for both MATLAB as well as ThingSpeak logins.)

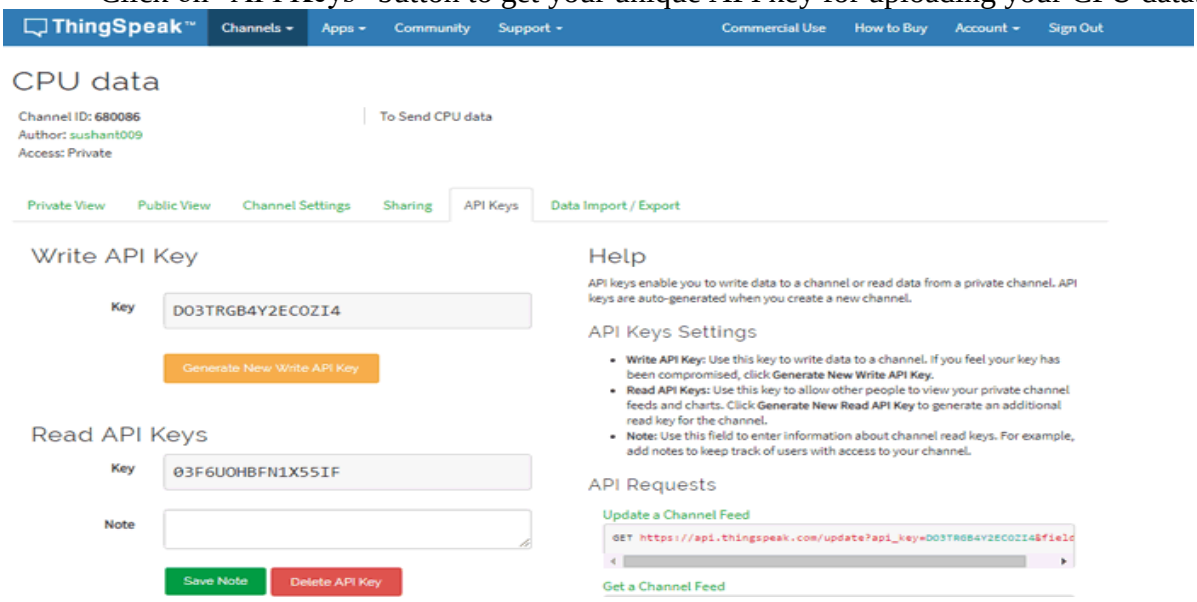
- After logging in, create a new channel for the data to be stored. For this go to Channels–>My Channels and click on New Channel.
- Enter the name of the channel and name of Field 1 in the corresponding sections. Fields in a channel are used to hold the data and each channel can have up to 8 fields. After entering the details save the channel.

- For example, a Channel called “Test Channel” is created and the Field 1 as “Random Number” as shown below.

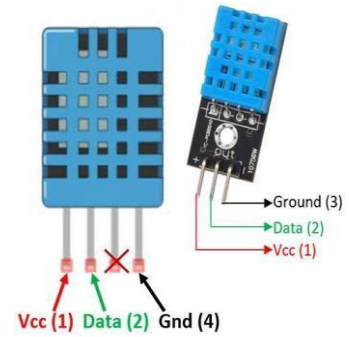


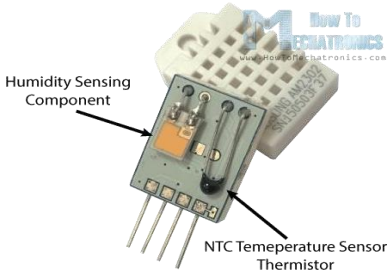
Getting API Key in ThingSpeak

- To send data to ThingSpeak, we need an unique API key, which is used to upload sensor data to ThingSpeak Website.
- Click on “API Keys” button to get your unique API key for uploading your CPU data.

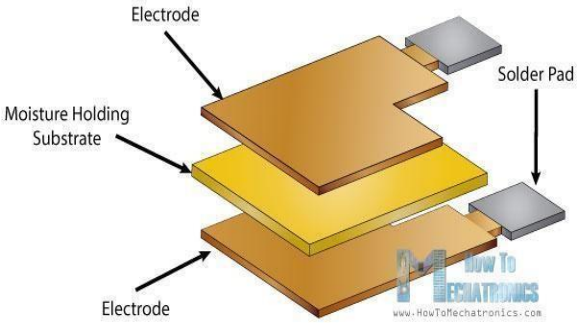


2. DHT11: DHT11 Temperature & Humidity Sensor features a temperature & humidity sensor complex with a calibrated digital signal output. By using the exclusive digital-signal-acquisition technique and temperature & humidity sensing technology, it ensures high reliability and excellent long-term stability. This sensor includes a resistive-type humidity measurement component and an NTC temperature measurement component, and connects to a high performance 8-bit microcontroller, offering excellent quality, fast response, anti-interference ability and cost-effectiveness.





2.2 DHT11 Working Principle: They consist of a humidity sensing component, a NTC temperature sensor (or thermistor) and an IC on the back side of the sensor.



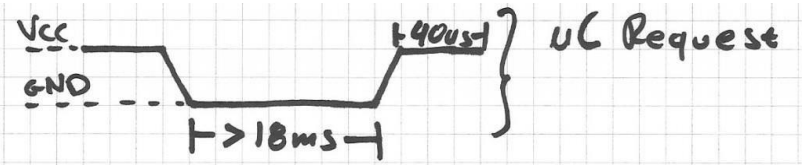
Humidity sensing component has two electrodes with moisture holding substrate between them. As the humidity changes, the conductivity of the substrate changes or the resistance between these electrodes changes. Change in resistance is measured & processed by the IC which makes it ready to be read by a microcontroller. A thermistor is actually a variable resistor that changes its resistance with change of the temperature. These sensors are made by sintering of semiconductive materials such as ceramics or polymers to provide larger changes in the resistance with just small changes in temperature.

2.3 DHT11 Communication Process

- Serial Interface (Single-Wire Two-Way) is used to communicate with DHT11
- All the sensor readings are sent using a single wire bus which reduces the cost and extends the distance.
- Communication Format can be separated into three stages
    - Request
    - Response
    - Data Reading

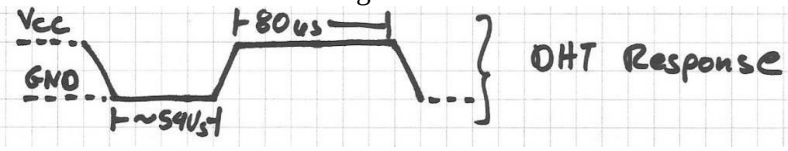
2.3.1 Communication Process: Request

- To make the DHT-11 to send you the sensor readings you have to send it a request.
- The request is, to pull down the bus for more than 18ms in order to give DHT time to understand it and then pull it up for 40uS.



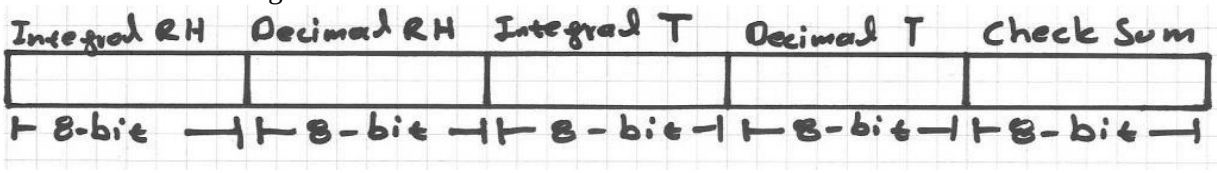
2.3.2 Communication Process: Response

- What comes after the request is the DHT-11 response.
- This is an automatic reply from DHT which indicates that DHT received your request.
- The response is ~54uS low and 80uS high.



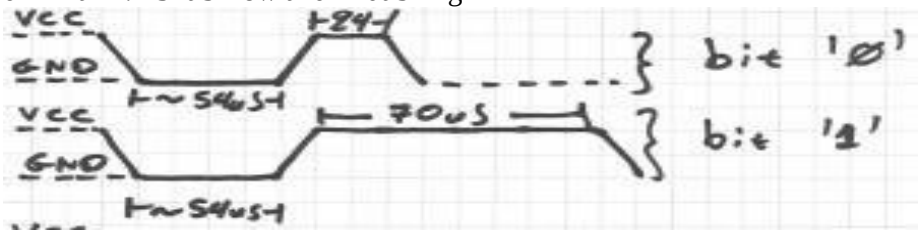
2.3.3 Communication Process: Data Read

- The sensor data after response will be packed in a packet of 5 segments of 8-bits each.
    - 2 segments are Humidity read, integral & decimal.
    - 2 segments are Temperature read in Celsius, integral & decimal
    - last segment is the Check Sum which is the sum of the 4 first



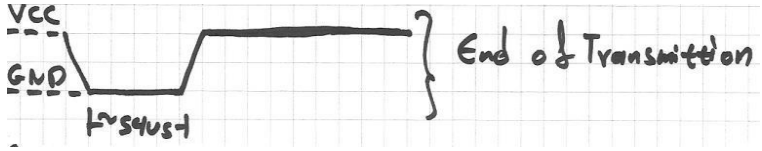
2.3.4 How to Identify Bits

- Each bit sent is a follow of ~54uS Low in the bus and ~24uS to 70uS High depending on the value of the bit.
    - Bit '0' : ~54uS Low and ~24uS High
    - Bit '1' : ~54uS Low and ~70uS High



2.3.5 End Of Frame

- At the end of packet DHT sends a ~54uS Low level, pulls the bus to High and goes to sleep mode.



- Example Reading
    - Humidity 0b00101011.0b00000000 = 43.0%
    - Temperature 0b00010111. 0b00000000 = 23.0 C.

3. Adafruit DHT11 Libarary Installation

This library is used for the accessing the DHT11, DHT22 readings. Download the library:

sudo pip3 install adafruit-circuitpython-dht

sudo apt-get install libgpiod2

4. Usage of DHT library

- Include the DHT11 library package and board package
    - import board
    - import adafruit_dht
- Declare the type of DHT i.e. DHT11 or DHT22 and Pin
    - dhtDevice = adafruit_dht.DHT11(board.D19)
    - In above D19 is GPIO19 pin to which the DHT11 is connected
- Call the reading function:
    - temperature = dhtDevice.temperature
    - humidity = dhtDevice.humidity
- Print the result on Terminal

  print("Humidity = {:.2f}%\tTemperature ={:.2f}C" .format(humidity, temperature))

5. Sending the data on Thingspeak

```
import time
import board
import adafruit_dht
from urllib.request import urlopen
import sys

WRITE_API = " 86GXULODML8R1GD9" # Replace your ThingSpeak API key here
BASE_URL = "https://api.thingspeak.com/update?api_key={}".format(WRITE_API)


# Initial the dht device, with data pin connected to:
dhtDevice = adafruit_dht.DHT11(board.D19, use_pulseio=False)

if__name__== '_main_':
   while True:
      try:
         # Print the values to the serial port
         temperature = dhtDevice.temperature
         humidity = dhtDevice.humidity
         print("Temp: {:.1f} C Humidity: {}%".format(temperature, humidity))
         thingspeakHttp = BASE_URL + "&field1={:.2f}&field2={:.2f}".format(temperature,
humidity)
         print(thingspeakHttp)
         conn = urlopen(thingspeakHttp)
         print("Response: {}".format(conn.read()))
         conn.close()
         time.sleep(15)

      except RuntimeError as error:
         # Errors happen fairly often, DHT's are hard to read, just keep going
         print(error.args[0])
         time.sleep(1.0)
         continue

      except KeyboardInterrupt:
         conn.close()
         print ('Exiting Program')
         exit()
```

Thonny - /home/pi/Desktop/New/MQTT_EFC3.py @ 12 : 13

New Load Save Run Debug Over Into Out Stop Zoom Quit Switch to regular mode

MQTT_EFC3.py

```
import time
import board
import adafruit_dht
from urllib.request import urlopen
import sys

WRITE_API = "86GXULODML8R1GD9" # Replace your ThingSpeak API key here
BASE_URL = "https://api.thingspeak.com/update?api_key={}".format(WRITE_API)
# Initial the dht device, with data pin connected to:
dhtDevice = adafruit_dht.DHT11(board.D19, use_pulseio=False)
if __name__ == '__main__':
    while True:
        try:
            # Print the values to the serial port
            temperature = dhtDevice.temperature
            humidity = dhtDevice.humidity
            print("Temp: {:.1f} C Humidity: {}%".format(temperature, humidity))
            thingspeakHttp = BASE_URL + "&field1={:.2f}&field2={:.2f}".format(temperature, h
            print(thingspeakHttp)
            conn = urlopen(thingspeakHttp)
            print("Response: {}".format(conn.read()))
            conn.close()
            time.sleep(15)
```

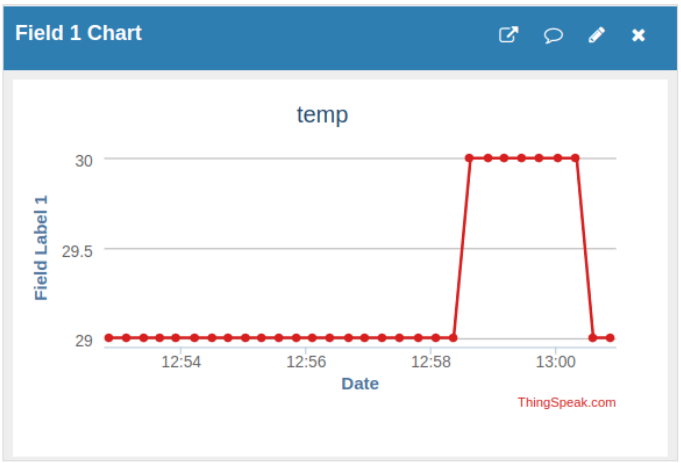
Shell

```
Temp: 29.0 C Humidity: 82%
https://api.thingspeak.com/update?api_key=9TQC9G088SNKXZ6L&field1=29.00&field2=82.00
Response: b'34'
Temp: 29.0 C Humidity: 83%
https://api.thingspeak.com/update?api_key=9TQC9G088SNKXZ6L&field1=29.00&field2=83.00
Response: b'35'
A full buffer was not returned. Try again.
Temp: 29.0 C Humidity: 83%
https://api.thingspeak.com/update?api_key=9TQC9G088SNKXZ6L&field1=29.00&field2=83.00
Response: b'36'
Temp: 29.0 C Humidity: 83%
https://api.thingspeak.com/update?api_key=9TQC9G088SNKXZ6L&field1=29.00&field2=83.00
Response: b'37'
```

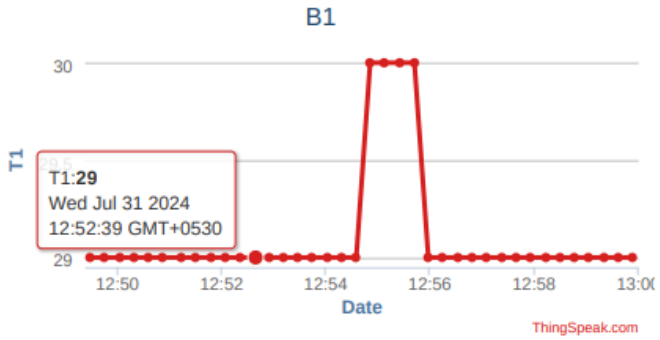
Python 3.9.2

thingspeak.com/channels/2612198/private_show

ThingSpeak™



31/07/2024, 13:00 https://thingspeak.com/channels/2613352/charts/1?bgcolor=%23ffffff&color=%23d62020&dynamic=true...



31/07/2024, 13:01 https://thingspeak.com/channels/2613352/charts/2?bgcolor=%23ffffff&color=%23d62020&dynamic=true...