



メカトロニクスの日本フランス週間
JFWM 2024
JAPANESE FRENCH WEEK ON MECHATRONICS

2024.09.12^月 - 20^日_{FR}
@ ANNECY, FRANCE

Supports:



CLUB DES
ENTREPRISES
Université Savoie Mont Blanc



CAMPUS
DES MÉTIERS
ET DES
QUALIFICATIONS
D'EXCELLENCE
Mécanique, construction
et énergie, informatique
et cryptologie, sciences



Lecture and tutorial: “IoT modules and its tutorial by using ~~SBC~~ MCU”

Takayuki Fujita

Professor



AMERI
University of Hyogo
Himeji, JAPAN



Contents

- **Introduce myself**
- **Overview of IoT/MEMS**
- **Instruction for IoT tutorial**

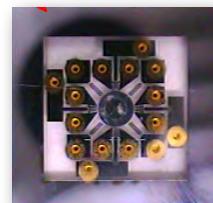
University of Hyogo

The screenshot shows the University of Hyogo website. At the top, there is a logo and the text "公立大学法人 兵庫県立大学 UNIVERSITY OF HYOGO". Below the logo, there are links for "ホーム" (Home), "English", "アクセスマップ" (Access Map), and "お問い合わせ" (Contact). There are also buttons for "文字サイズ 標準 大" (Text Size Standard Large) and "標準 大" (Standard Large). A navigation bar below has links for "受験生の方へ" (For Admissions), "在学生・教職員の方へ" (For Students and Staff), "卒業生の方へ" (For Graduates), and "社会人へ" (For Professionals). The main content area features a large image of a modern building, with the text "Graduate School of Engineering" in pink and "工学部 school of engineering" in white. A blue button says "工学部公式サイトはこちら" (Link to the Faculty of Engineering website). Below the image, there is Japanese text: "兵庫県立大学 工学部" and "柔軟な発想と個性豊かな独創力を發揮できる人材を養成します". The URL in the address bar is "http://www.uoh.ac.jp/faculty-of-engineering/".

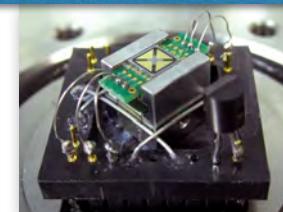


My research history

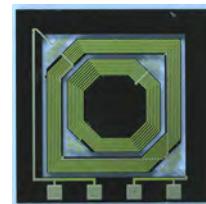
- MEMS physical sensors
 - Gyroscope
 - Accelerometers
- MEMS actuators
 - Mirror device
 - PID control system
- Environmental sensor
 - Multi sensor module for health monitoring



2D gyroscope



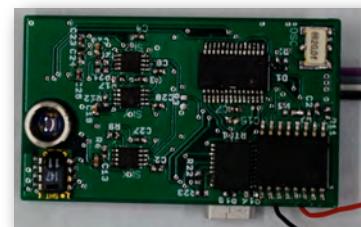
Vibratory beam accelerometer



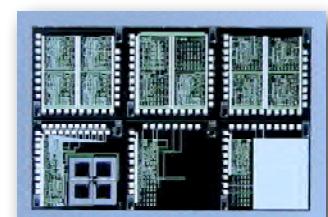
Electromagnetic MEMS mirror



Mirror control system



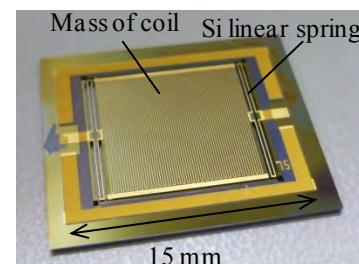
Multi-sensor module



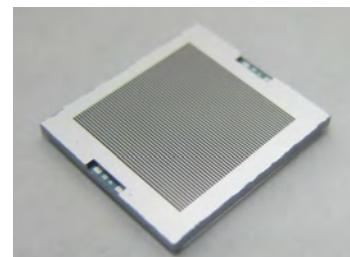
Environmental sensor

Recently, working on

- **Vibration MEMS energy harvester**
- **IoT systems for Medical applications**



Vibration energy harvester



Clean room facilities for MEMS



CleanRoom for MEMS

- Operated & managed by
 - 3 faculties
 - 1 technician
 - ~20 students

- **Full MEMS fabrication process on site**
 - Photomask fabrication, photolithography
 - Deep-RIE, dry etching, wet etching
 - Metallization, Magnetic/PZT film sputtering
 - Electroplating, Polymer-flexible electronics

AMERI, U-Hyogo

**Move to New institute
open Apr. 2022**

**Advanced Medical Engineering
Research Institute,
University of Hyogo**



AMERI

Joint research facility



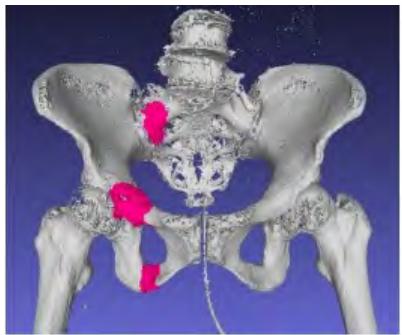
**兵庫県立
はりま姫路総合医療センター
Harima Himeji
General Medical
Center**

In the 2nd largest prefectural hospital in Hyogo

Hospital



Research Topics of AMERI



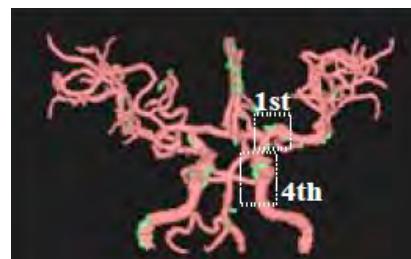
AI-based pelvic
fracture location
determination



Health observation using
adhesive plaster-type
MEMS sensors



Finger joint
estimation robotic
prosthetic hand



AI-based aneurysm
visualization



Silicone flexible 3D printed
artificial heart valve

Ex.1) Tube self-extraction sensing

Unscheduled removal of tubes is a serious problem



Did you ou pull it out?



Sensor monitors
> Pull out
> Signs of Pulling

Two types of capacitive sensor



Stretchable force sensor
With adhesive touch sensor + BLE beacon module
Battery

Medical + IoT

Removal of tube monitoring on Cloud

Realtime visualization by Grafana



**Grafana is very powerful visualization tool
We'll have tutorial with Grafana later**

Ex.2) Fall down detection around bed

The number of incidence of falls among elderly patients in hospital is high



- **Falls** account for **20.5%** of incidents in Japan.*
(total falls: 983, total incidents: 4802)
- Half of the falls occur at bed-leaving.

It is difficult to frequently observe patients due to staff shortages

Bed-leaving sensors are being increasingly utilized as preventive measures against falls during bed-leaving behavior.

* Japan Council for Quality Health Care, Medical Accident Information Collection and Related Activities, Annual Report 2020

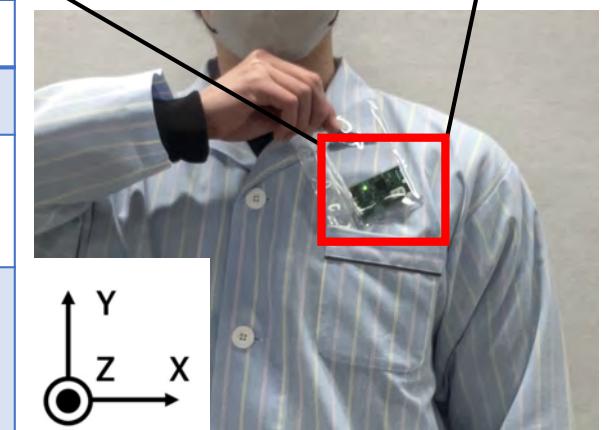
9-DOF Sensor and BLE MCU

STEVAL-MKSBOX1V1TM product summary	
iNEMO 6DoF inertial module	LSM6DSOX
Digital 3-axis magnetometer	LIS2MDL
Digital nano pressure sensor	LPS22HH
Other features	BLE etc.

STEVAL-MKSBOX1V1™
(STMicroelectronics)

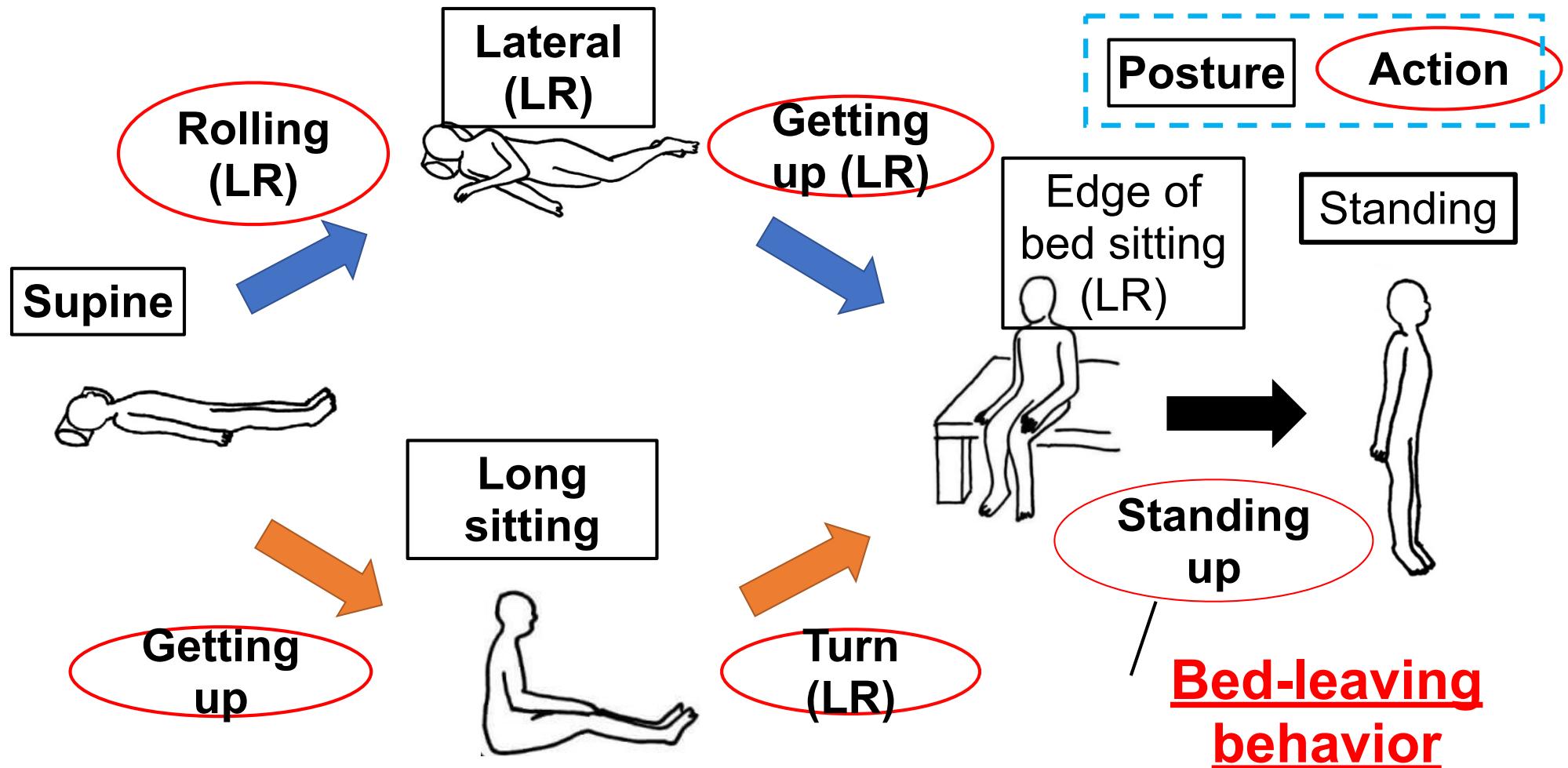


LSM6DSOX product summary	
Power consumption	0.55mA
full-scale acceleration range	$\pm 2g / \pm 4g / \pm 8g / \pm 16g$
full-scale angular rate range	$\pm 125\text{dps} / \pm 250\text{dps} / \pm 500\text{dps} / \pm 1000\text{dps} / \pm 2000\text{dps}$
MLC decision tree	Up to 8 trees, 32 features
Other features	Equipped with a MEMS motion sensor that incorporates MLC etc.



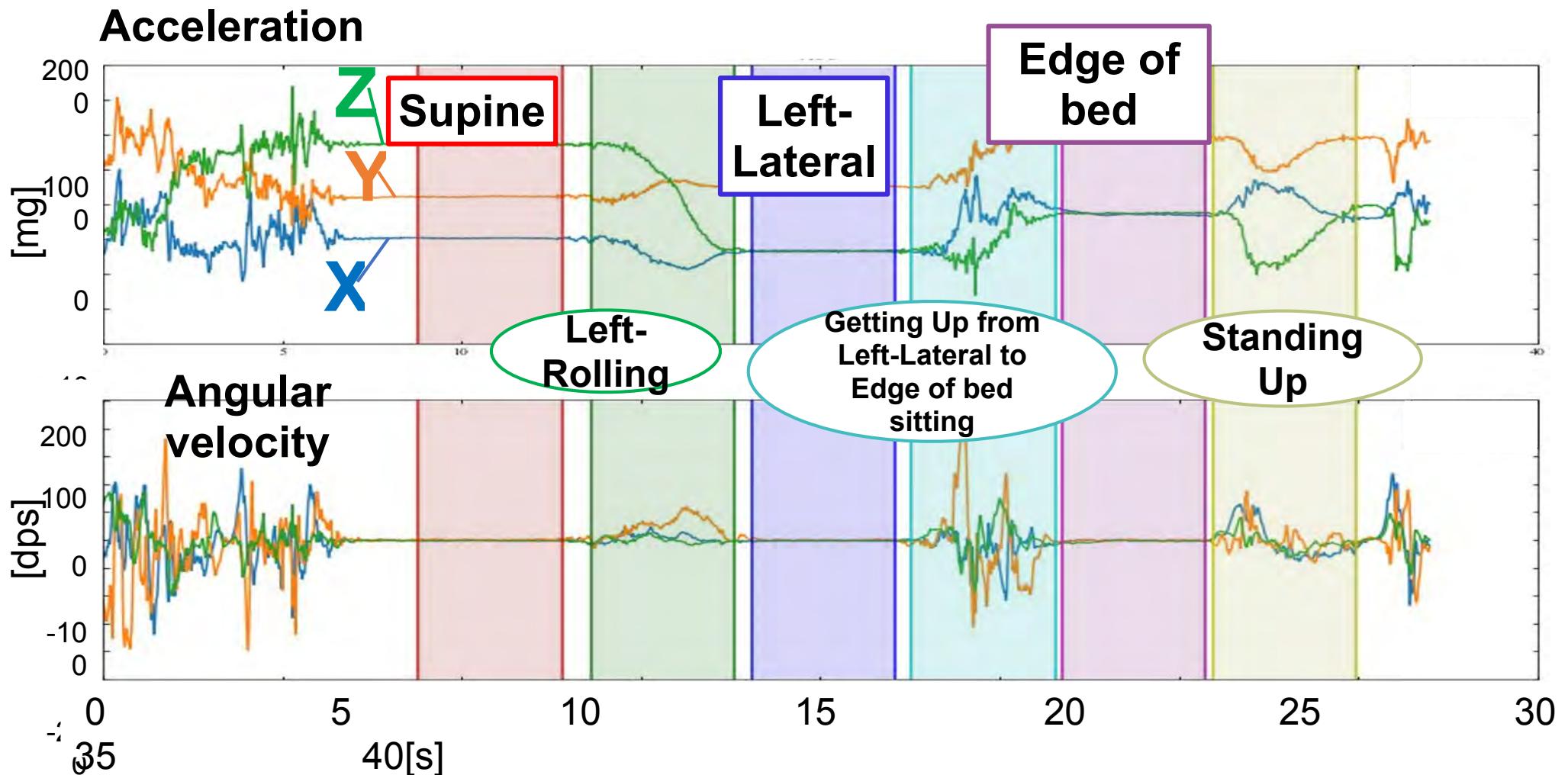
- Insert sensor into the chest pocket of the patient's clothing.

Postures and actions measurements



Training data preparation with predefined protocols

Annotation to Acc.& A.Velo. Data from situations



Real-time detection experiment



Value	Behavior
0	Supine
1	Supine to Left lateral
2	Supine to Right lateral
3	Left lateral
4	Right lateral
5	Supine to Long sitting
6	Long sitting
7	Left lateral to Left edge of bed
8	Right lateral to Right edge of bed
9	Long sitting to Left edge of bed
10	Long sitting to Right edge of bed

Medical + IoT + AI

Contents

- Introduce myself
- **Overview of IoT/MEMS**
- Instruction for IoT tutorial

IoT (Internet of Things)



IoT

Internet connection for everything not limited to Human operated device

Human uses the Internet

Smartphone,
Tablet,
PC

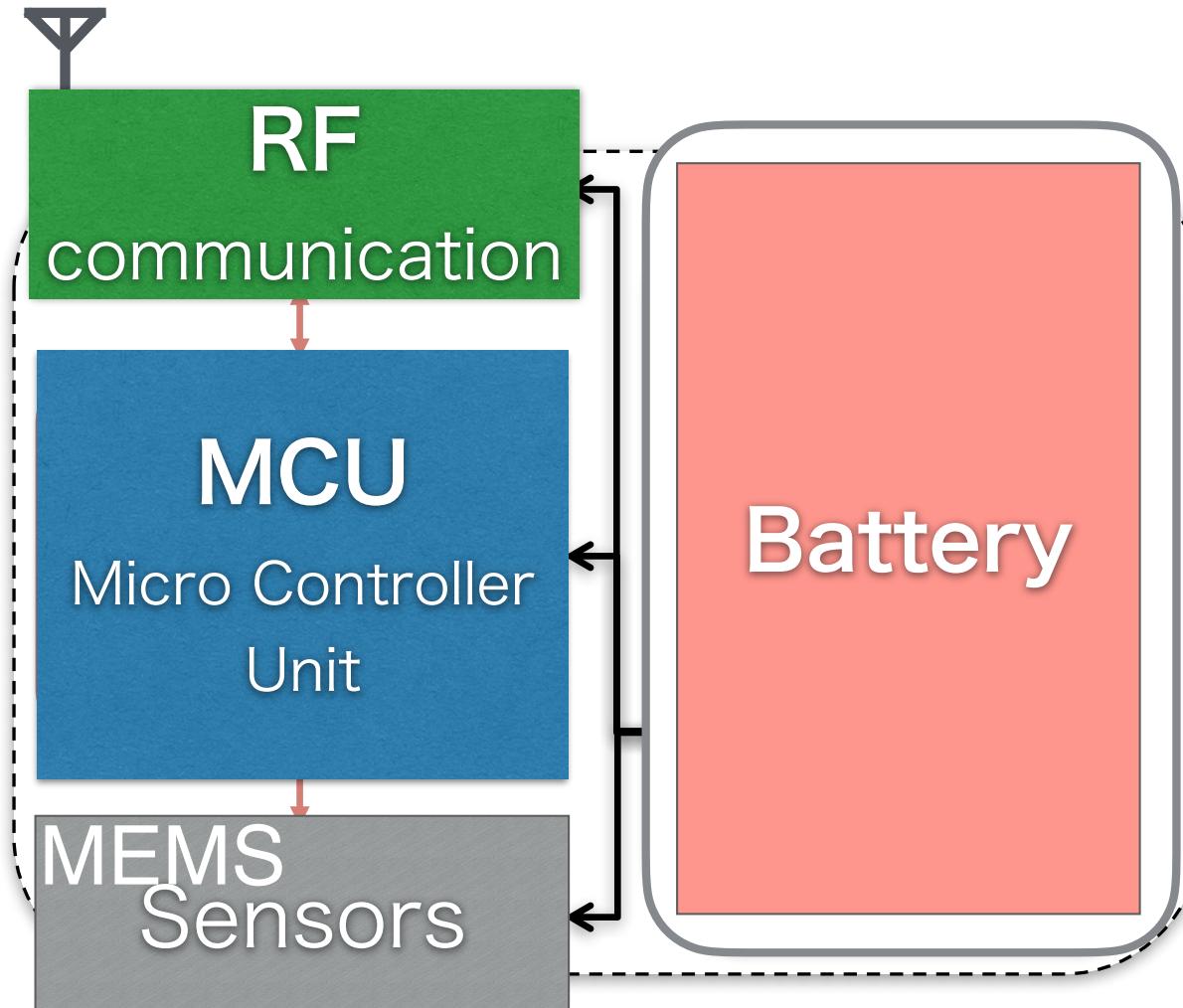


Lots of sensors connect autonomously



IoT node architecture

Typical architecture of IoT node



Architecture of a sensor node

IoT has developed dramatically thanks to **MEMS**

MEMS - Micro Electromechanical Systems

**MEMS = Fusion of ultra small machine
and Integrated Circuitry (IC)**

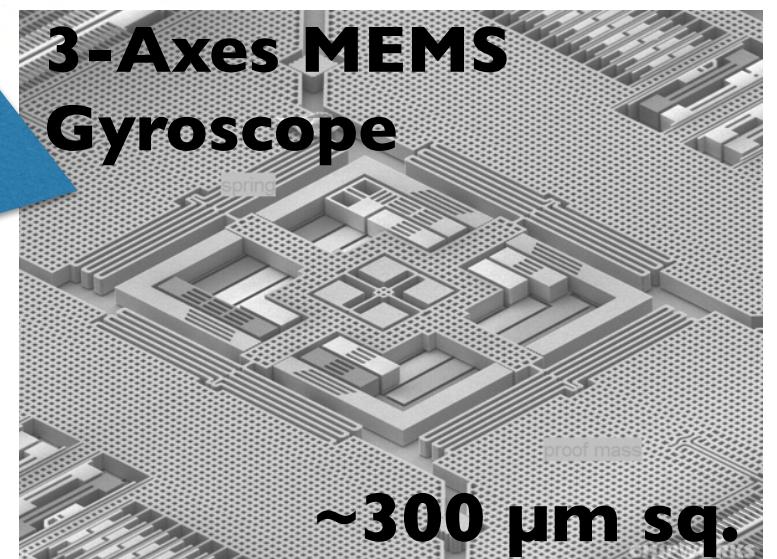
- IMU combo
- Magnetometer
- MEMS microphones
- Pressure sensor
- Humidity + Temperature sensor
- BAW filters and duplexers
- Antenna tuner



**Full of MEMS
sensors in smart
phone**

at least 5 MEMS sensors

Inside?



History of MEMS application



Automotive



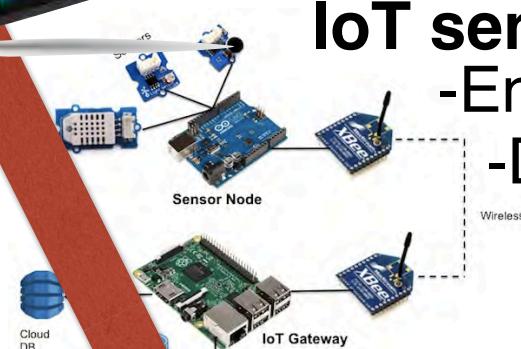
- Airbag control
- Electronic stability control
- Rollover detection
- Navigation (GPS support)



Mobilephone / Entertainment



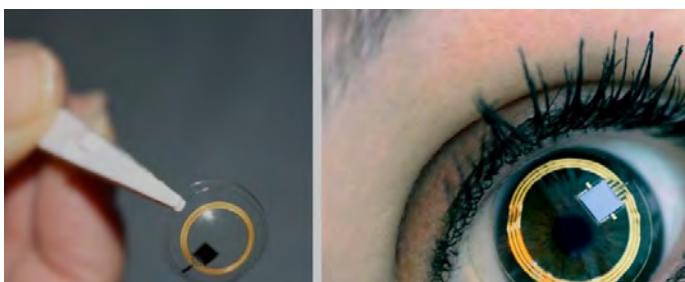
- Input device
- Navigation



IoT sensor node

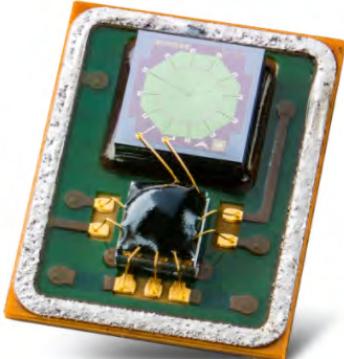
- Environmental sensing
- Drones

Health -Medical devices



Performance
Size & Cost
↓ Number ↑

MEMS device 1: Microphone



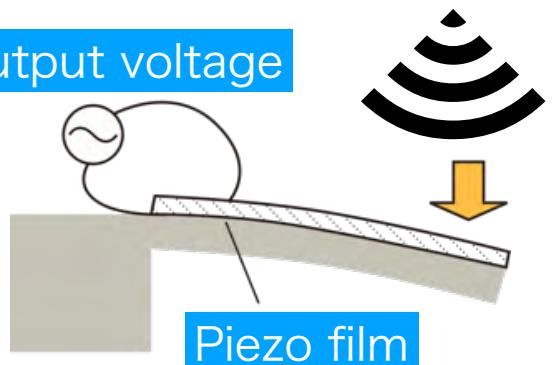
Rice grain size
 4 x 3 mm
© Vesper



Rice grain size

Cantilever with piezoelectric material vibrate by sound

Output voltage

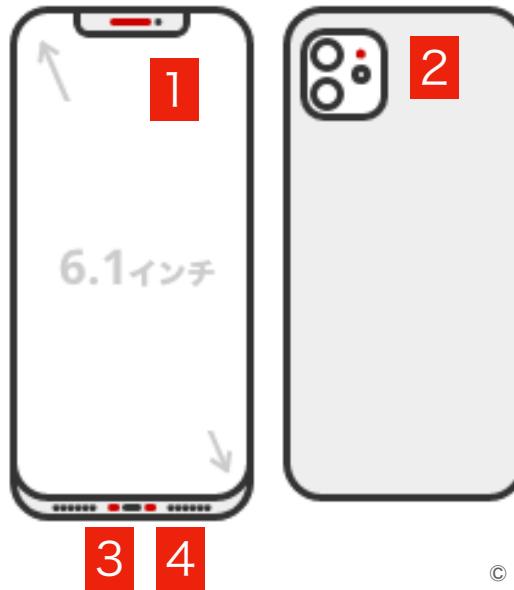


Sound wave

Piezo film

MEMS makes compact, high-performance microphones inexpensive

iPhone 11



© Voista Media

iPhone 11 has 4 microphones
3 4 for main mic.
Others are use for noise cancelation

MEMS device 2: Apple watch etc.

Activity tracking for health care



Various devices of “healthy gadgets”



Similar technology

RF, MCU, battery
&

MEMS sensors
Acc. Gyro,

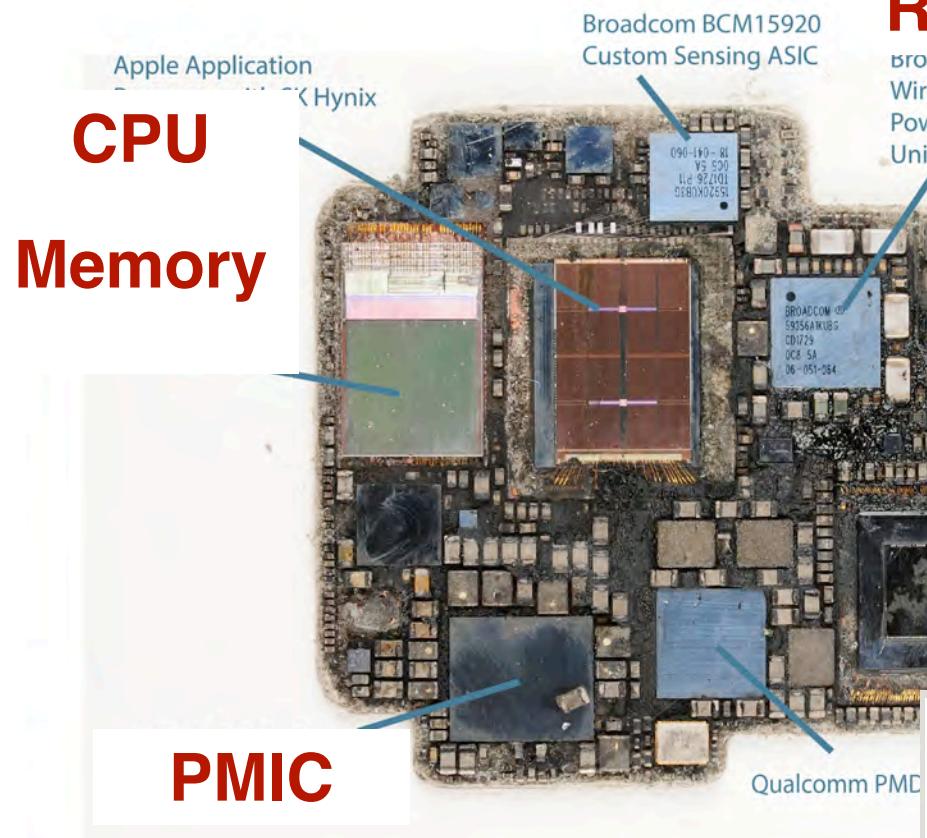
Optical sensor
For SPO₂, heart rate

- Heart rate sensor
- Accelerometer
- Gyroscope
- Bluetooth 4.0 LE



Inside of AppleWatch

(C) TechInsights



RF/wireless

Broadcom BCM159350
Wireless Charging
Power Management
Unit

Apple 338S00348V
chip (likely)

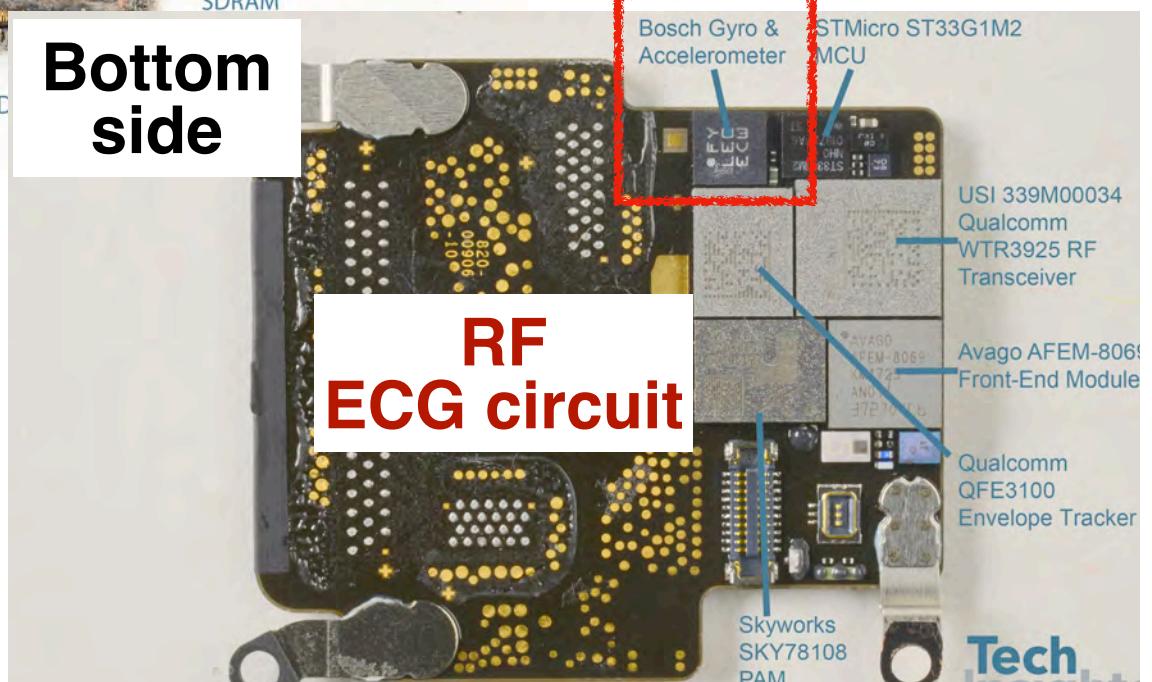
Top side

Bottom side

AppleWatch 3rd gen.



MEMS
Gyroscope +
Accelerometer



Tech

IoT on market : IoT Body scale

¥20,000
(USD 150)
worth the price

Withings

Made in France

Smart Body Analyzer

US\$ 149.95

- WiFi connection
- No switch, Easy to use
- Automatically data upload

Pinpoint your weight and body composition

Get high-accuracy weight and body fat measurement as well as your Body Mass Index. Pick the right body type to optimize the fat mass measurement – if you exercise vigorously on a regular basis, switch to "Athlete mode".



Weight

2011 2012 2013 2014 2015 2016 THIS YEAR

Weight (kg)

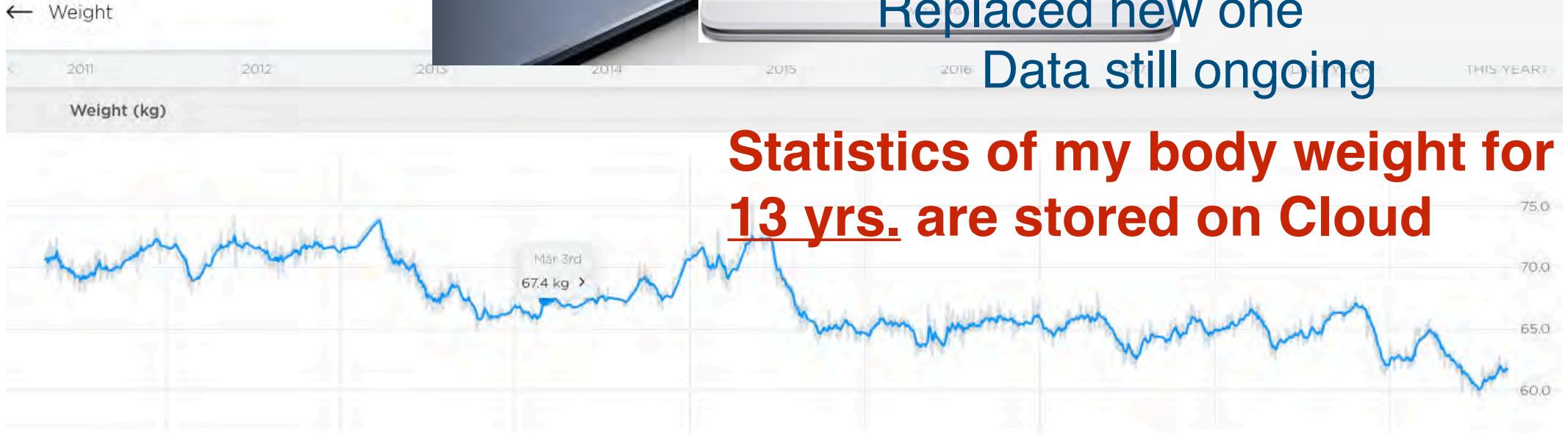
Mar 3rd 67.4 kg

Replaced new one

Data still ongoing

Weight BMI Fat Mass Heart rate

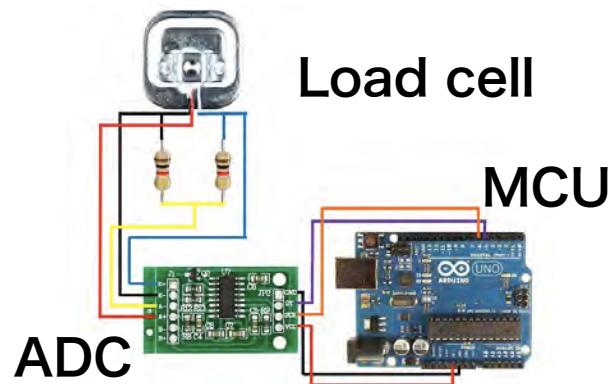
Statistics of my body weight for
13 yrs. are stored on Cloud



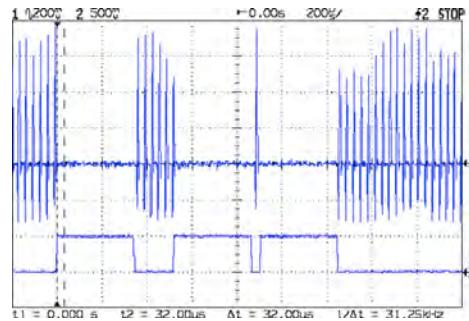
Inside of Withings IoT scale



**Body weight measured
by load-cell (strain sensor)**



Body fat % by impedance measurement



Withings server

wi+things

my account

Don't have an account yet?

mail address

メールアドレス

password

パスワード

I forgot the password?

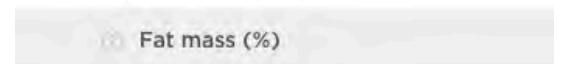


← Weight

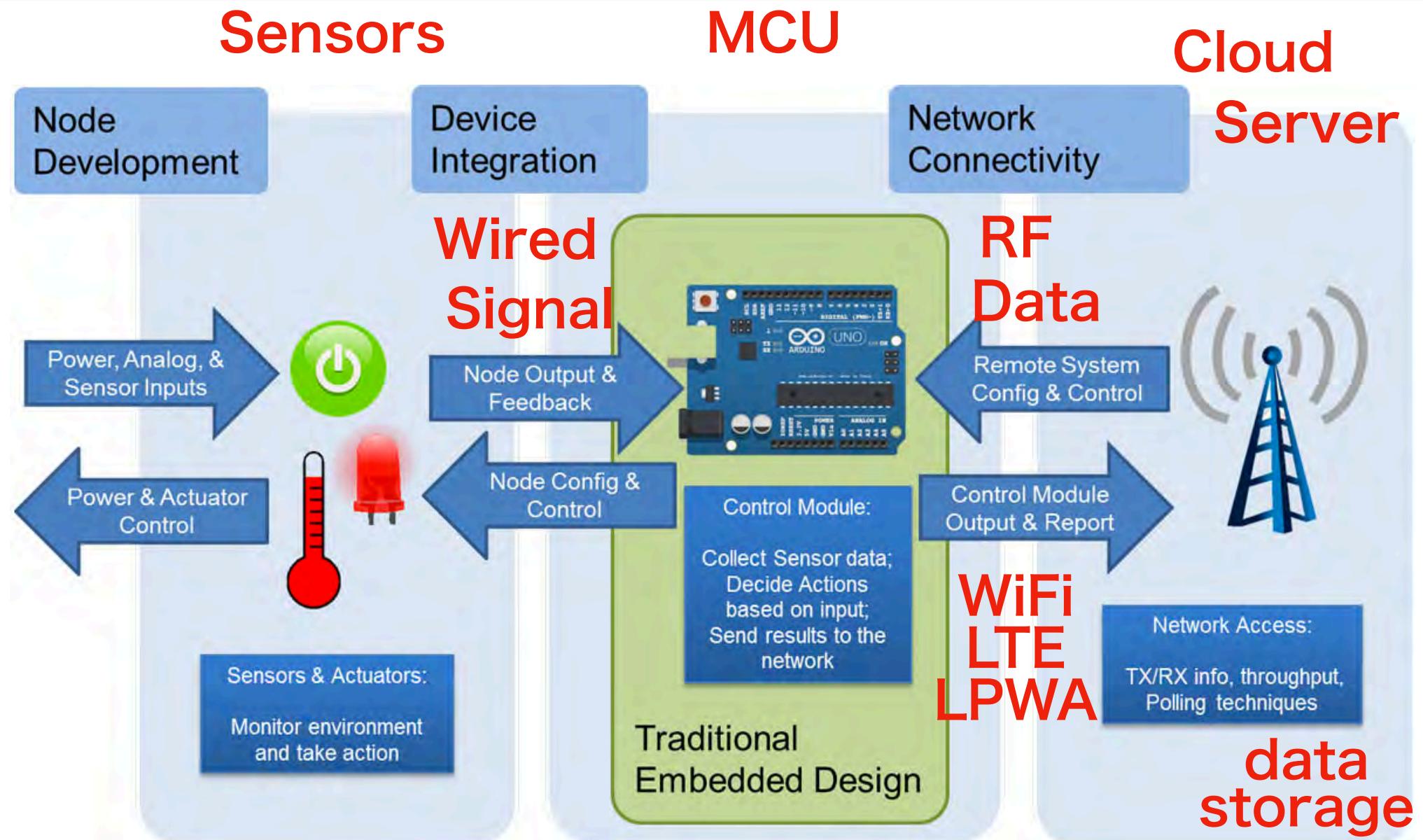
**Generate
graph
on cloud**



**Connect via Wifi (pre-configured SSID / password)
> upload data to Withings cloud server**



Typical configuration IoT



>NEXT

Introduction to the history of measurement,
using temperature as an example

Temperature measurement (primitive method)

ex. Glass tube thermometer

Physical change
> Temperature



Have you ever seen it?

Physical change
> thermal expansion of alcohol



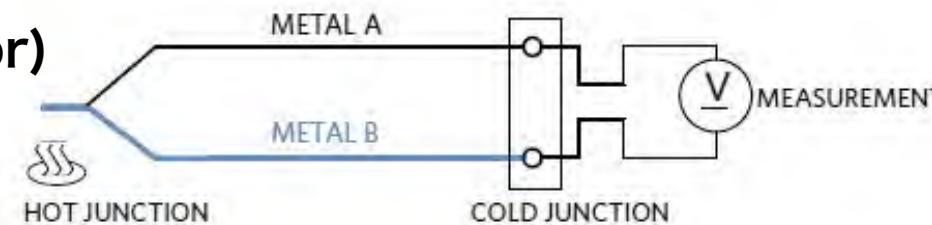
No electricity

Visual check only

ex. Thermocouple
(temperature sensor)



Physical change
> Temperature



Electrical change
> Electromotive force



Analog meter



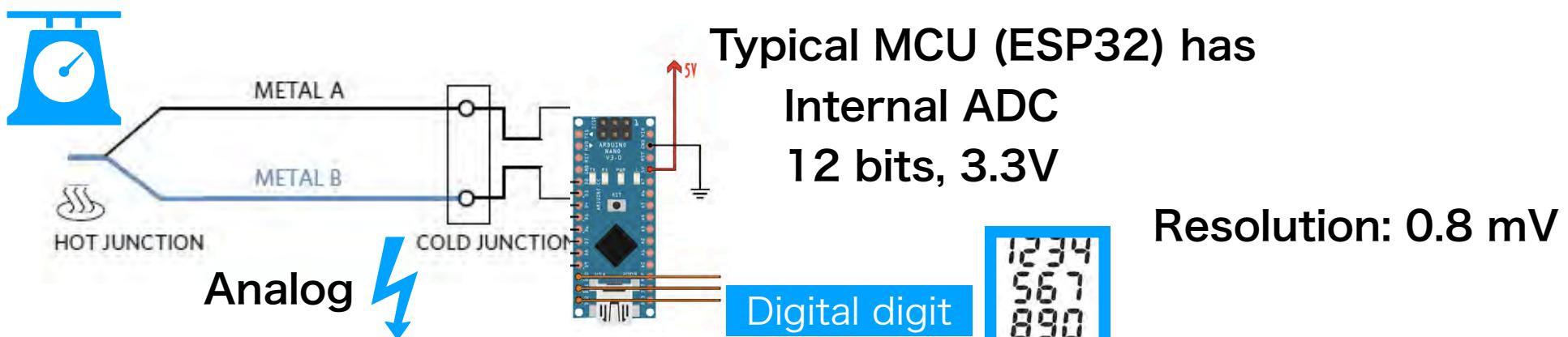
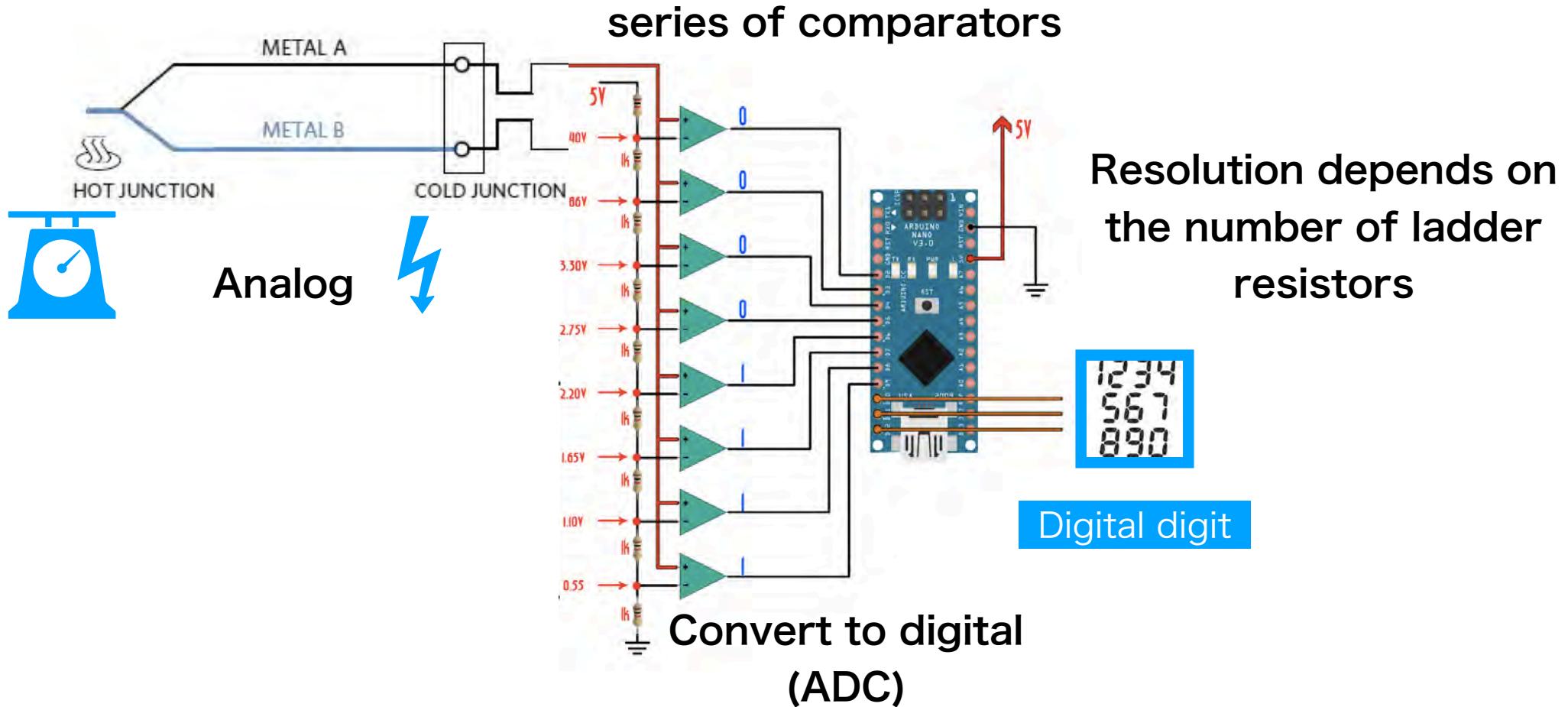
Analog value

Physical change
> meter movement



Analog values are difficult to connect MCU or SBC

Temperature measurement (w/ ADC)



Recent digital output MEMS sensor



BOSCH

Invented for life

BME280
combo sensor



Use this on tutorial

Fully digital output
with compensation
parameters

Single chip
-Temperature
-Humidity
-Pressure

Main features



Relative Humidity
Measures relative humidity
with a fast response time



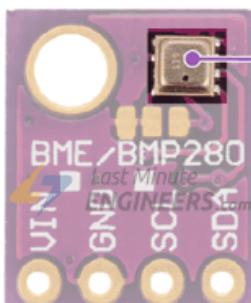
Pressure
Measures barometric
pressure and altitude



Temperature
Measures ambient
temperature



Digital



Sensor



BME280 Chip

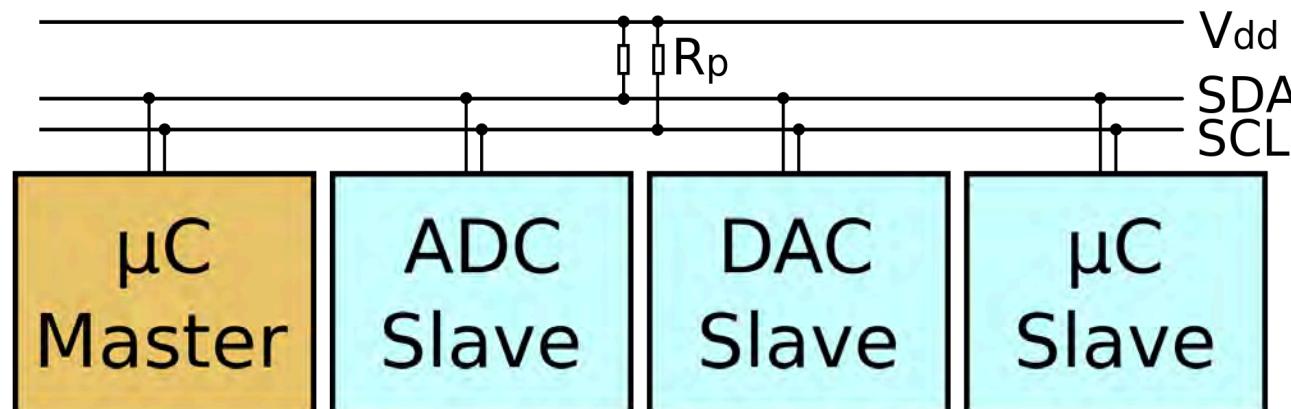
Direct connection
via I₂C/SPI protocol

What is interface
protocol?

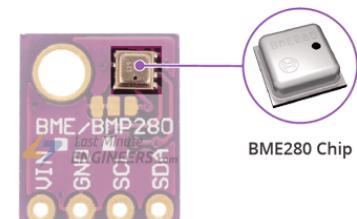
1) Inter-Integrated Circuit (I²C)



I²C (Inter-Integrated Circuit), pronounced *I-squared-C*, is a synchronous, multi-master, multi-slave, packet switched, single-ended, serial computer bus. It is widely used for attaching lower-speed peripheral ICs to processors and microcontrollers in short-distance, intra-board communication. Alternatively I²C is spelled **I2C** (pronounced I-two-C) or **IIC** (pronounced I-I-C).

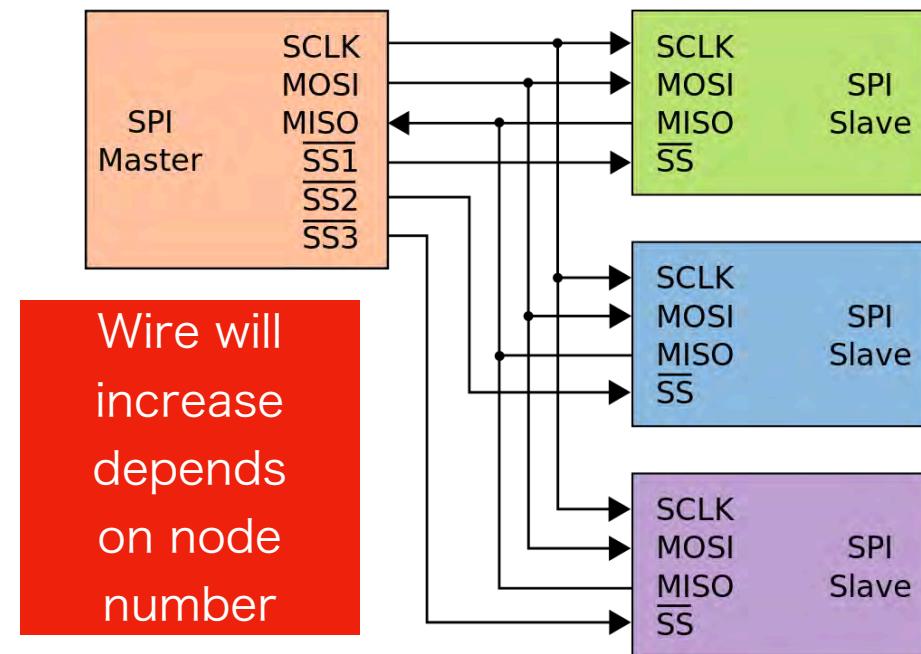
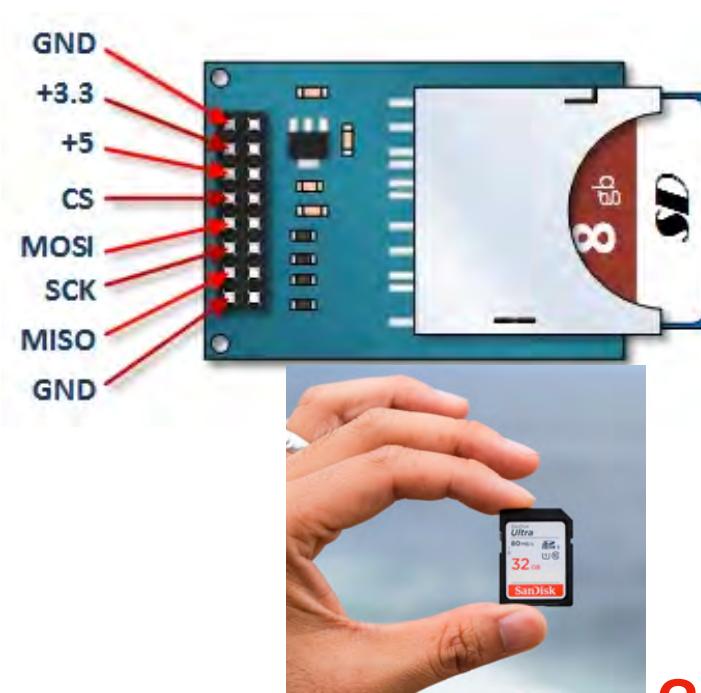


2 digital signal line + 2 power line
Selected by unique-**address**



2) Serial Peripheral Interface (SPI)

The **Serial Peripheral Interface (SPI)** is a **synchronous serial communication** interface specification used for short-distance communication, primarily in **embedded systems**. Typical applications include **Secure Digital cards** and **liquid crystal displays**. SPI devices communicate in **full duplex** mode using a **master-slave** architecture with a single master. The master device originates the **frame** for reading and writing. Multiple slave-devices are supported through selection with individual **slave select (SS)**, sometimes called chip select (CS), lines.



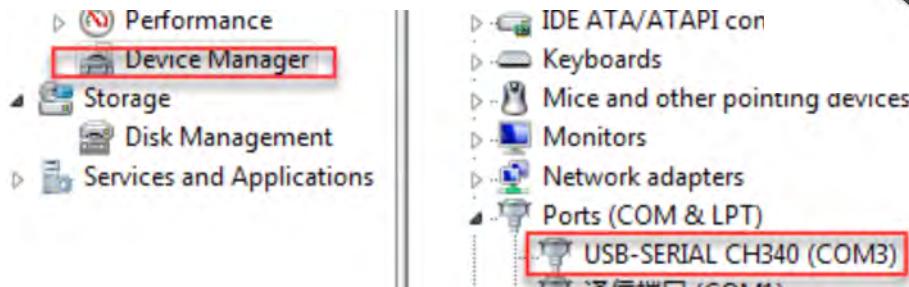
**2 digital signal line + 2 power line
+1 CS line for chip select**

3) Universal Asynchronous Receiver/Transmitter (UART)

A universal asynchronous receiver-transmitter (UART) is a computer hardware device for **asynchronous serial communication** in which the data format and transmission speeds are configurable. The electric signaling levels are handled by a driver circuit external to the UART. Two common signal levels are **RS-232**, a 12-volt system, and **RS-485**, a 5-volt system.



Old PC has **RS-232C**
Serial port

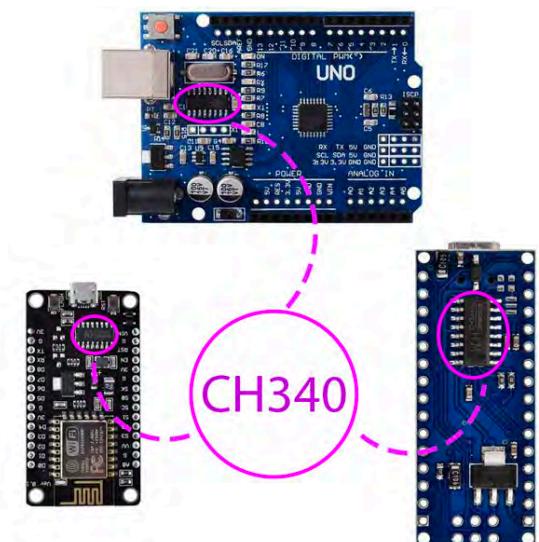


Windows recognize it as **COM**munication port

USB (Universal Serial Bus)
Replaced it



Almost all MCU board
includes USB-UART
converter chip
Ex. CH340



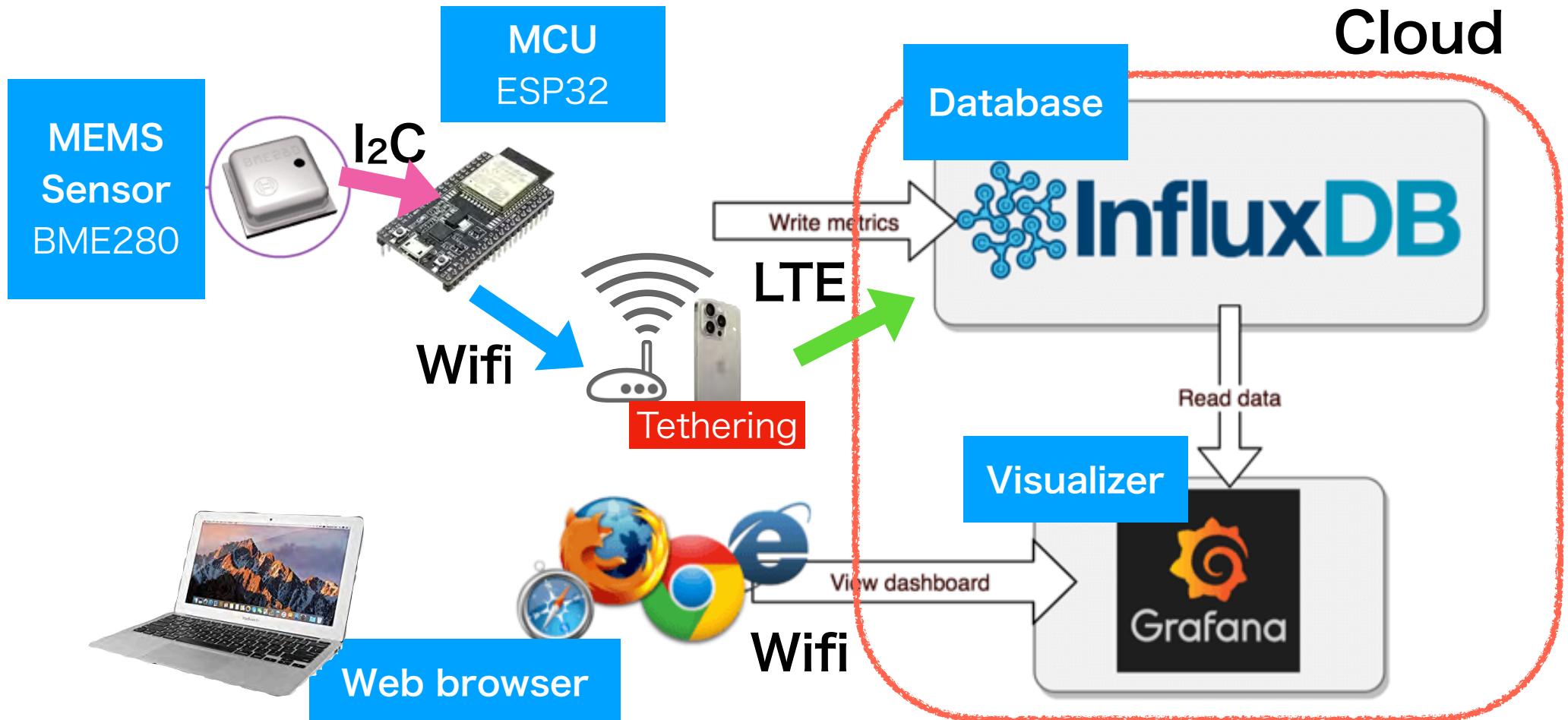
Contents

- **Introduce myself**
- **Overview of IoT/MEMS**
- **Instruction for IoT tutorial**

Goal of tutorial

Sensing data from **MEMS** environmental sensor

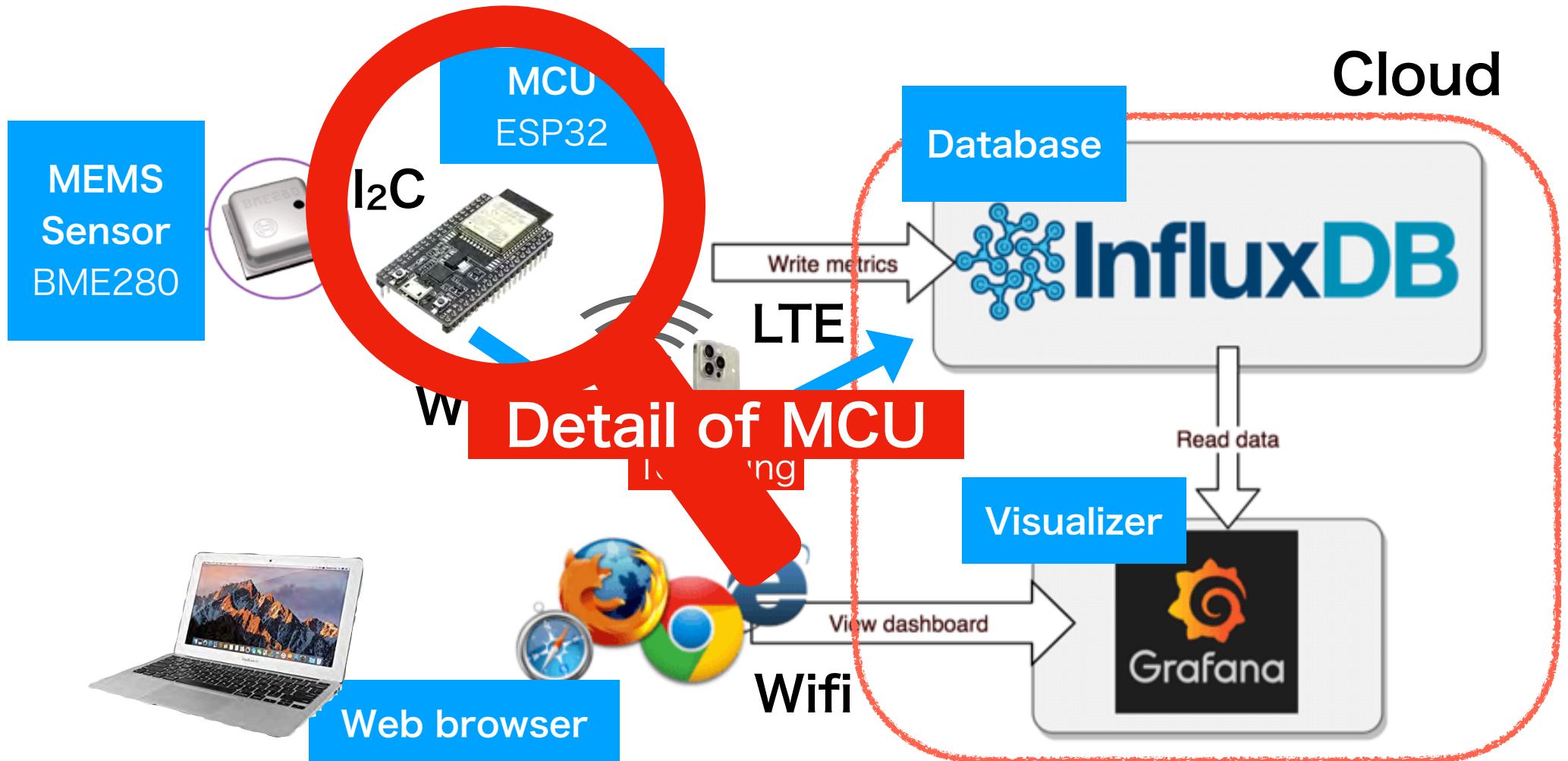
- > gather & processing data on **MCU**
- > transmit to **InfluxDB** database on cloud
- > visualized by **Grafana** > display on web browser



Goal of tutorial

Sensing data from **MEMS** environmental sensor

- > gather & processing data on **MCU**
- > transmit to **InfluxDB** database on cloud
- > visualized by **Grafana** > display on web browser



Wide variety of MCU variations

ARM-based MCU is popular these days

Apple silicon also
ARM based core

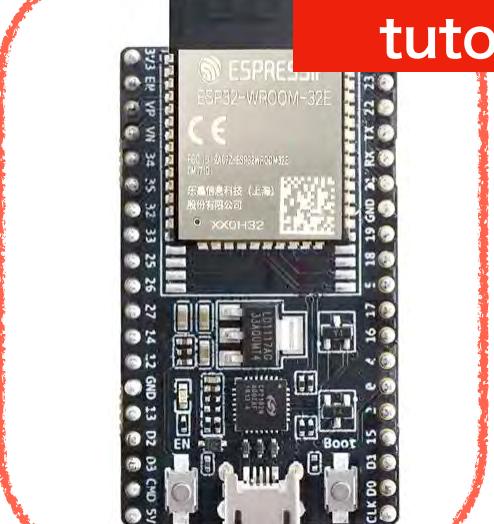
Raspberry Pi Pico



ARM Cortex M0+

ESP32

Use this on
tutorial



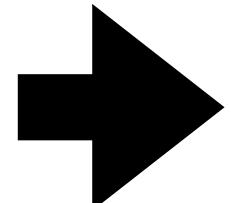
nRF52840



ARM Cortex M4

Each company's original IDE
(Integrated Development Environment)

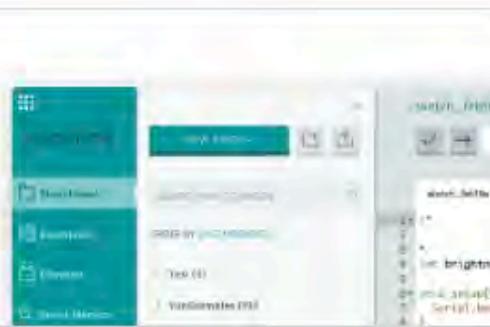
Generic development environment is
useful in terms of asset utilization





Arduino Web Editor

Start coding online and save your sketches in the cloud. The most up-to-date version of the IDE includes all libraries and also supports new Arduino boards.

[CODE ONLINE](#)[GETTING STARTED](#)

Originally Arduino Development Environment Downloads

But libraries Support Various MCUs



Arduino IDE 2.2.1

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocomplete, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

Nightly builds with the latest bugfixes are available through

Based on C/C++ language

DOWNLOAD OPTIONS

Windows Win 10 and newer, 64 bits

Windows MSI installer

Windows ZIP file

Linux AppImage 64 bits (X86-64)

Linux ZIP file 64 bits (X86-64)

macOS Intel, 10.14. "Mojave" or newer, 64 bits

macOS Apple Silicon, 11. "Big Sur" or newer, 64 bits

[Release Notes](#)

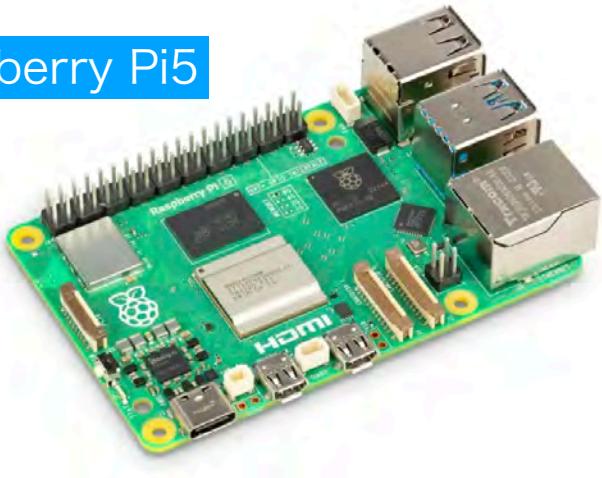
Help

Comparison of SBC and MCU

SBC

Single Board Computer
= Like a small PC

Raspberry Pi5



MCU

Micro Controller Unit
= CPU + peripheral I/F

ESP32



Very flexible operation

No OS
Single software = Firmware
e.g. Temp. Cont.



Very robust operation

This tutorial uses Python on the MCU as an alternative solution

The image shows a screenshot of the MicroPython website. At the top, there is a navigation bar with the MicroPython logo, DOWNLOAD, DOCS, DISCORD, DISCUSSIONS, WIKI, and STORE links. Below the navigation bar, the word "MicroPython" is written in large, bold, black letters. To the right of the title, there is a 3D rendering of a black microcontroller board (pyboard) resting on a white surface. Above the board, three colored boxes are stacked vertically: a blue box labeled ".py script", a pink box labeled "MicroPython", and a green box labeled "Hardware". On the left side of the page, there is descriptive text about MicroPython, its pyboard, its features, and its compatibility.

MicroPython is a lean and efficient implementation of the [Python 3](#) programming language that includes a small subset of the Python standard library and is optimised to run on microcontrollers and in constrained environments.

The MicroPython [pyboard](#) is a compact electronic circuit board that runs MicroPython on the bare metal, giving you a low-level Python operating system that can be used to control all kinds of electronic projects.

MicroPython is packed full of advanced features such as an interactive prompt, arbitrary precision integers, closures, list comprehension, generators, exception handling and more. Yet it is compact enough to fit and run within just 256k of code space and 16k of RAM.

MicroPython aims to be as compatible with normal Python as possible to allow you to transfer code with ease from the desktop to a microcontroller or embedded system.

Programming language for small resources compatible with Python

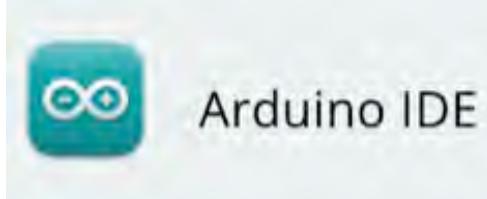
Run within just 256k of code space and 16k of RAM.

C++ IDE vs. μ Python (development)



C++

On PC



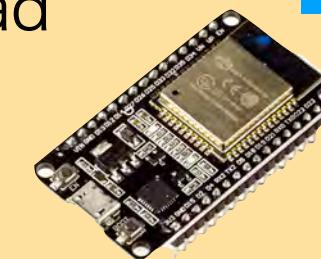
Arduino IDE

Programming & Compiling

Generate binary

Upload

On MCU



Reset, Operation
check

MicroPython

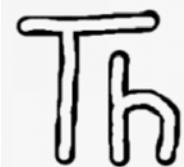


On MCU

REPL [Read-Evaluation-Print Loop]

Thonny IDE

On PC



Revise

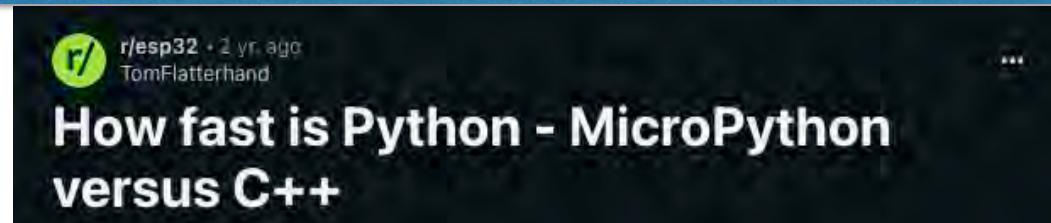
Once μ Python farm is uploaded



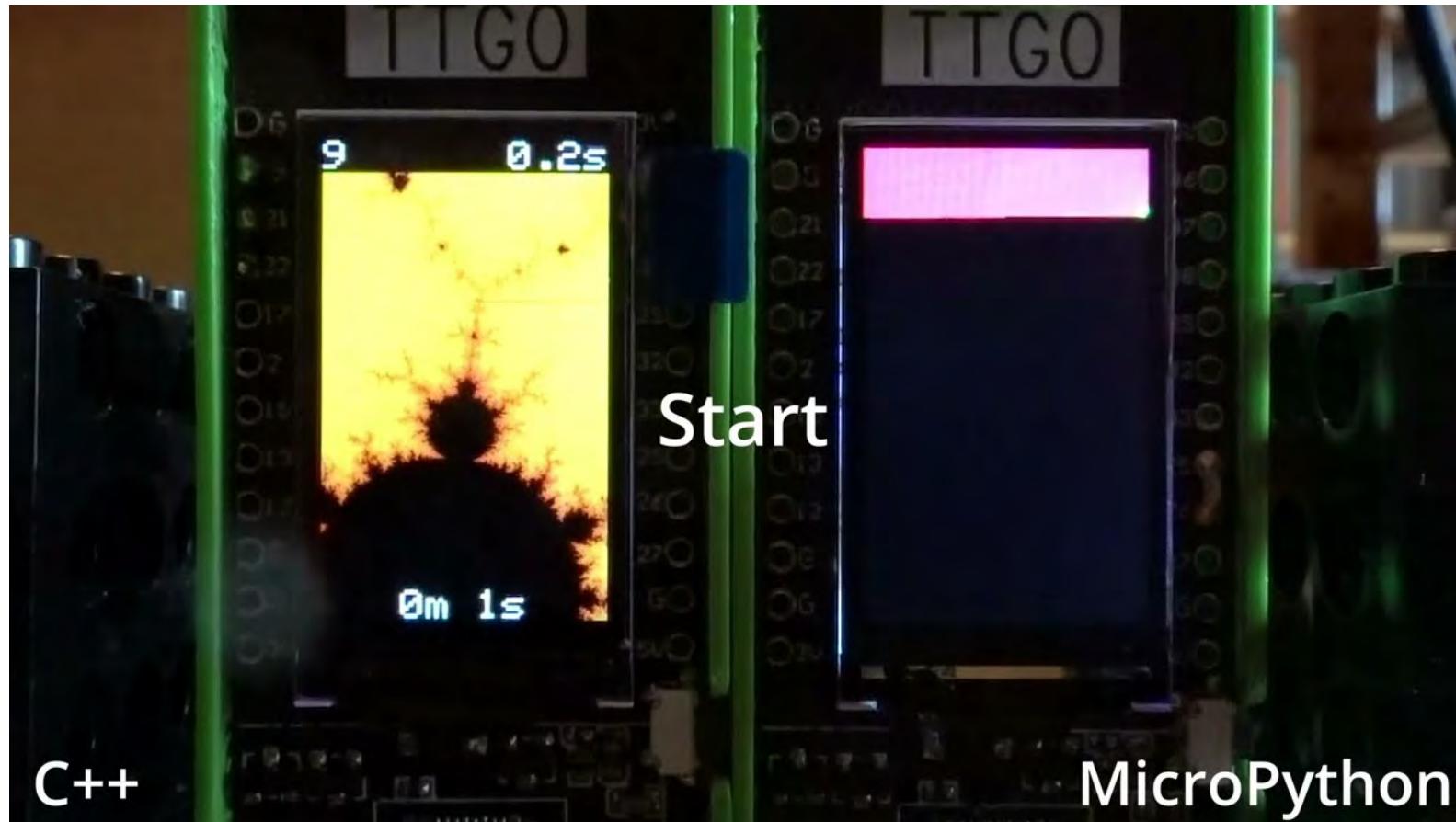
Operation check

**Very easy for
prototyping**

C++ IDE vs. μ Python (performance)



Mandelbrot with ESP32+LED



©YouTube

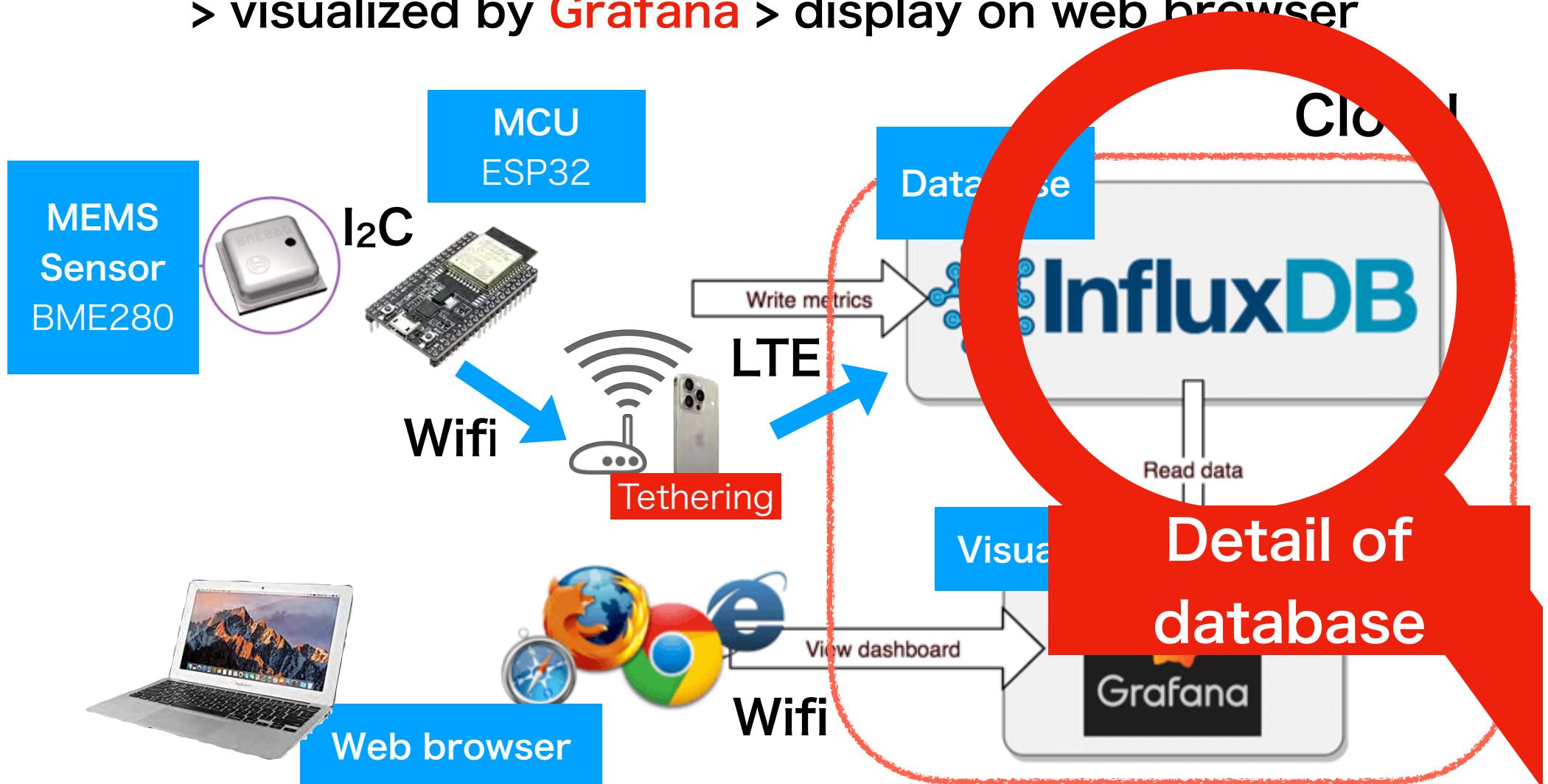
C++ is 400 times faster than μ Python @ ESP32

- > μ Python is for prototyping
- > C++ is for production

Goal of tutorial

Sensing data from **MEMS** environmental sensor

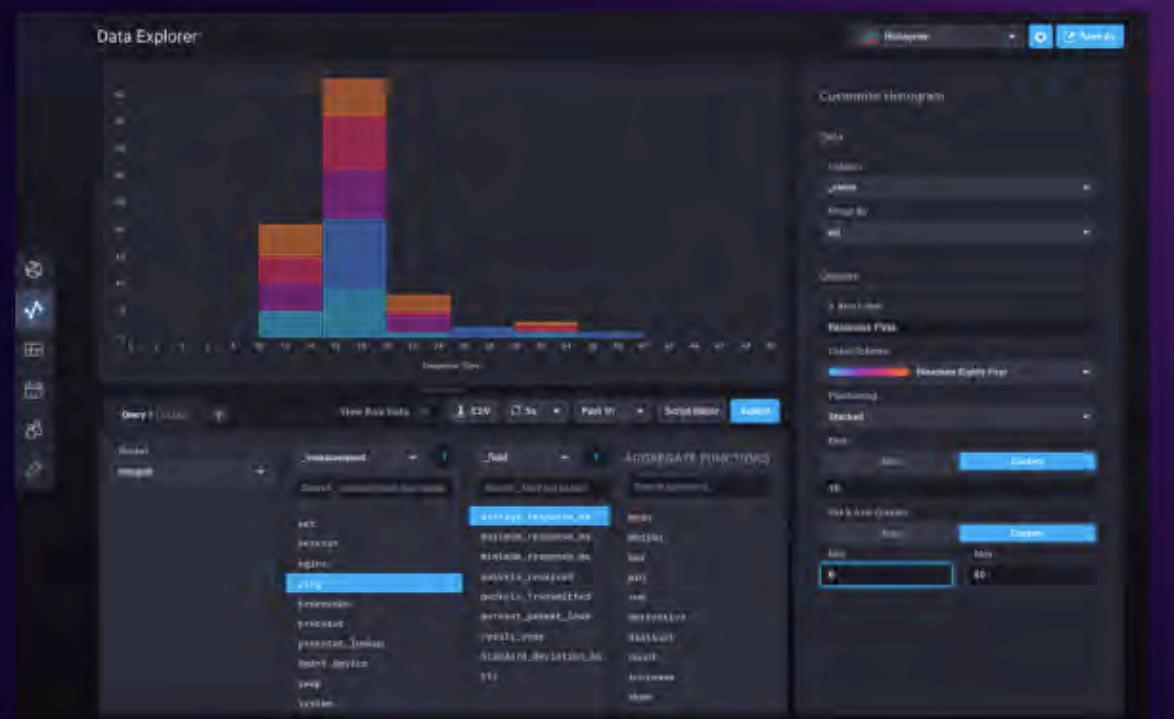
- > gather & processing data on **MCU**
- > transmit to **InfluxDB** database on cloud
- > visualized by **Grafana** > display on web browser



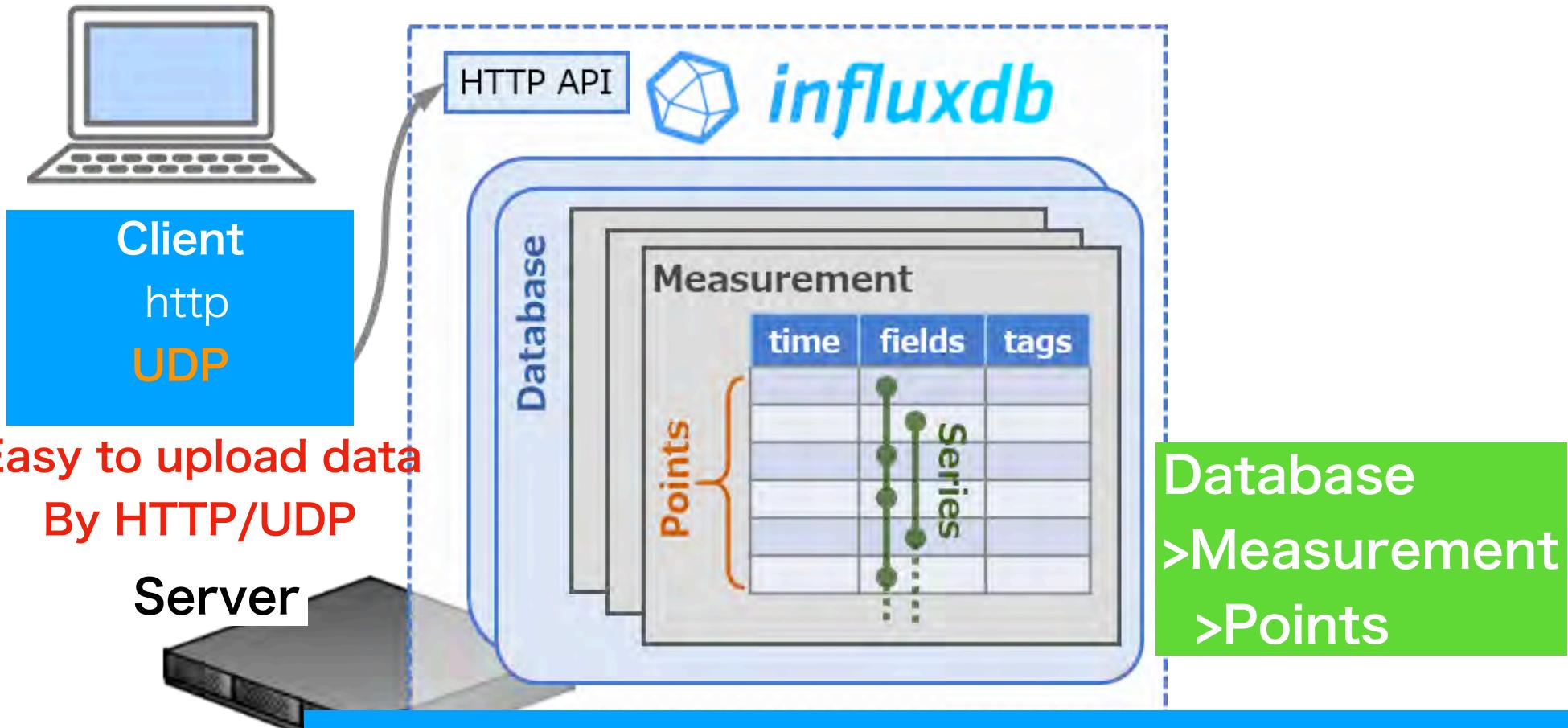
InfluxDB

InfluxDB is a time series database designed to handle high write and query loads.

[Get InfluxDB](#)



InfluxDB is a high-performance **data store** written specifically for **time series data**. It allows for high throughput ingest, compression and real-time querying. InfluxDB is written entirely in Go and compiles into a single binary with no external dependencies. It provides write and query capabilities with a command-line interface, **a built-in HTTP API**.



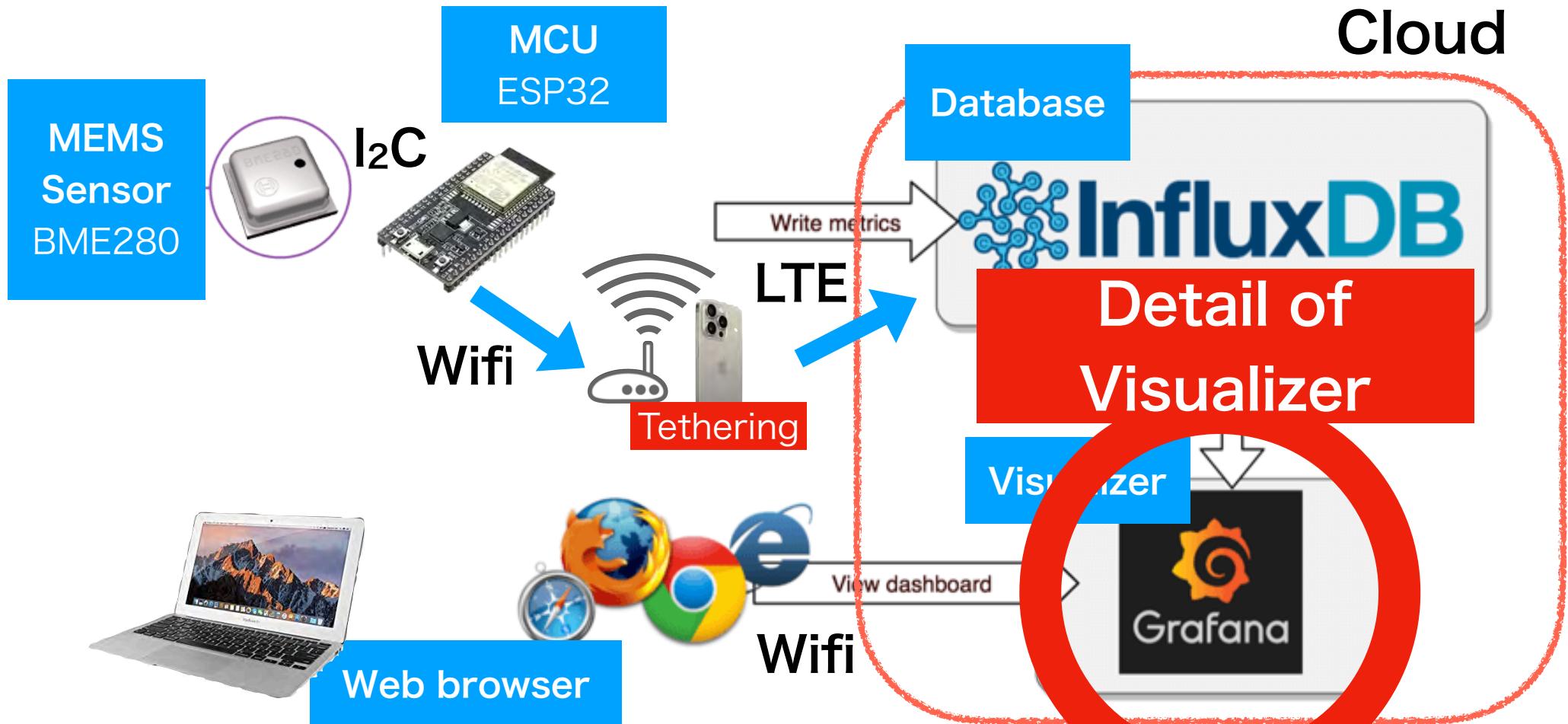
Structure of InfluxDB database

Database	Measurement		Values		
	Timestamp	Location	Temperature	Humidity	Barometer
Point	15977464390000000000	bedroom	23.4	67.8	1023.4
	15977464990000000000	bedroom	23.5	67.9	1023.6
	15977465590000000000	bedroom	23.5	68.1	1023.7

Goal of tutorial

Sensing data from **MEMS** environmental sensor

- > gather & processing data on **MCU**
- > transmit to **InfluxDB** database on cloud
- > visualized by **Grafana** > display on web browser



The leading **open source** software for time series analytics



No matter where your data is, or what kind of database it lives in, you can bring it together with Grafana. Beautifully.



[Learn More](#)

[Live Demo](#)

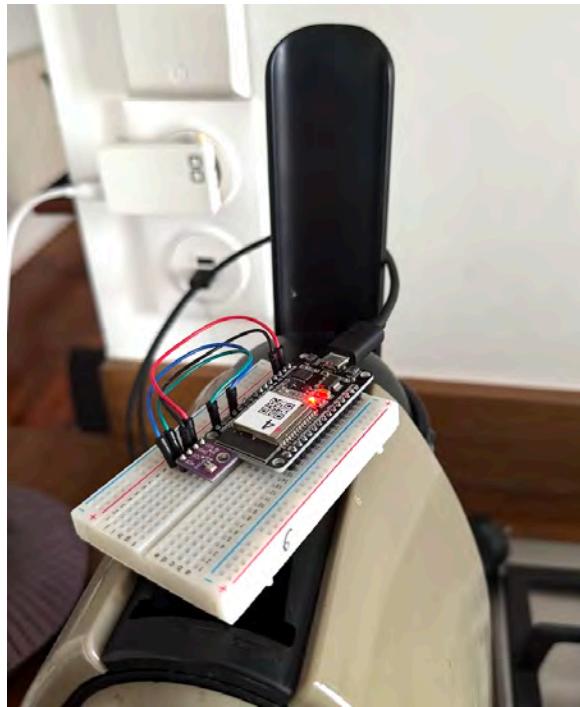
[Get Grafana](#)

Grafana allows you to query, **visualize**, alert on and understand your metrics no matter where they are stored. Create, explore, and share **dashboards** with your team and foster a data driven culture.

Example of Grafana (my home env.) on RPi



Tutorial IoT installed in my apartment where I'm staying.



Tutorial instruction



This afternoon

90' Lecture
10:00 - 11:30
T. Fujita, University of Hyogo, JP
IoT modules & its tutorial by using SBC
B014

Lunch break
11:30-13:15

90' Tutorial - 13:15 - 14:45
Y. Suzuki
Design of broadband vibration energy harvesting device
C114 - Group : B

90' Practical work
13:15 - 14:45
T. Fujita
IoT modules tutorial using SBC
C213 - Group : A

Challenge
13:15 - 14:45
Group : C

Challenge
15:00 - 16:30
Groups : A, B

A

C

Monday Sept 16

90' Lecture
8:15 - 9:45
H. Sawada
Image processing for Mechatronics
B120

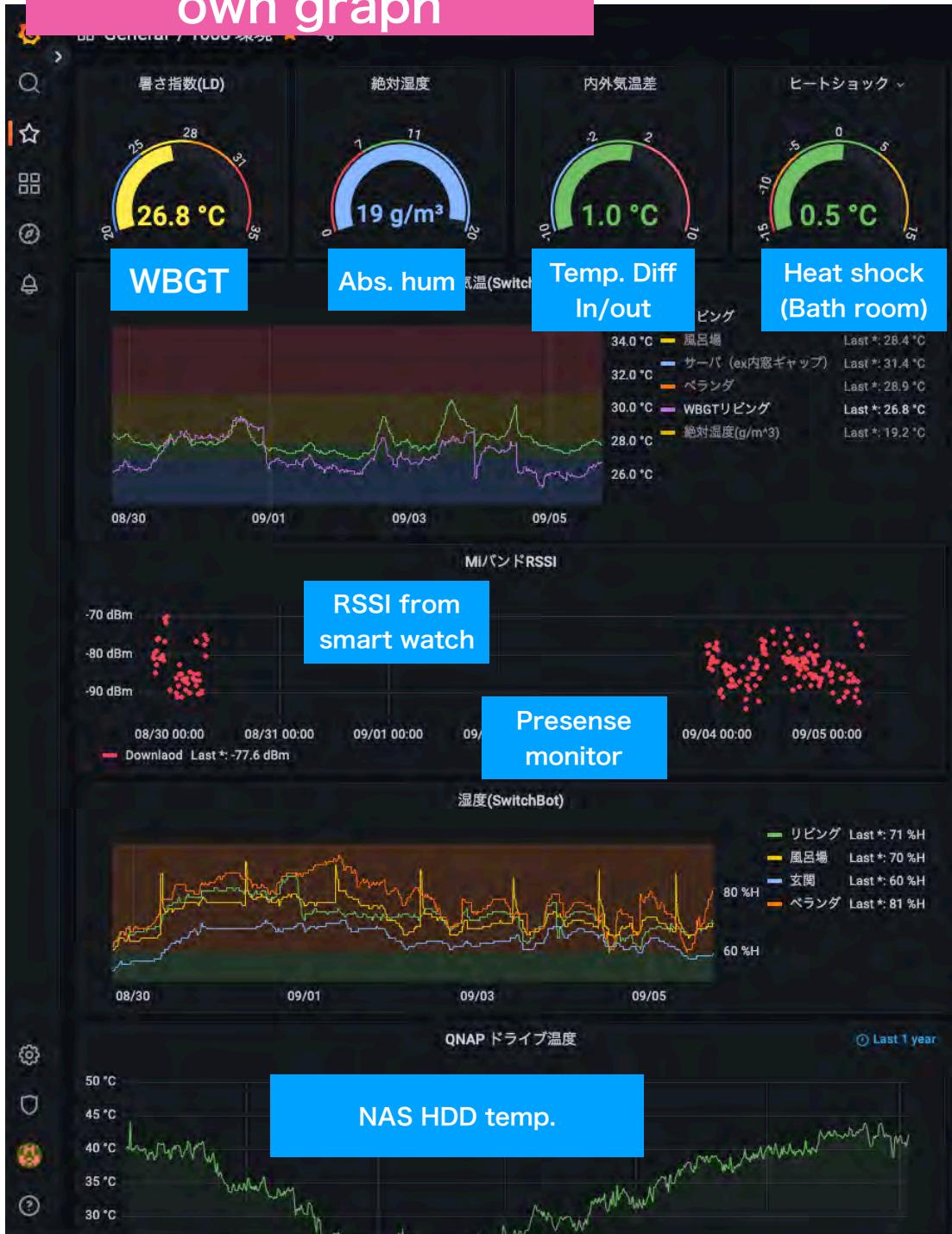
Coffe Break

90' Practical work
10:00 - 11:30
T. Fujita
IoT modules tutorial using SBC
C213 - Group : B

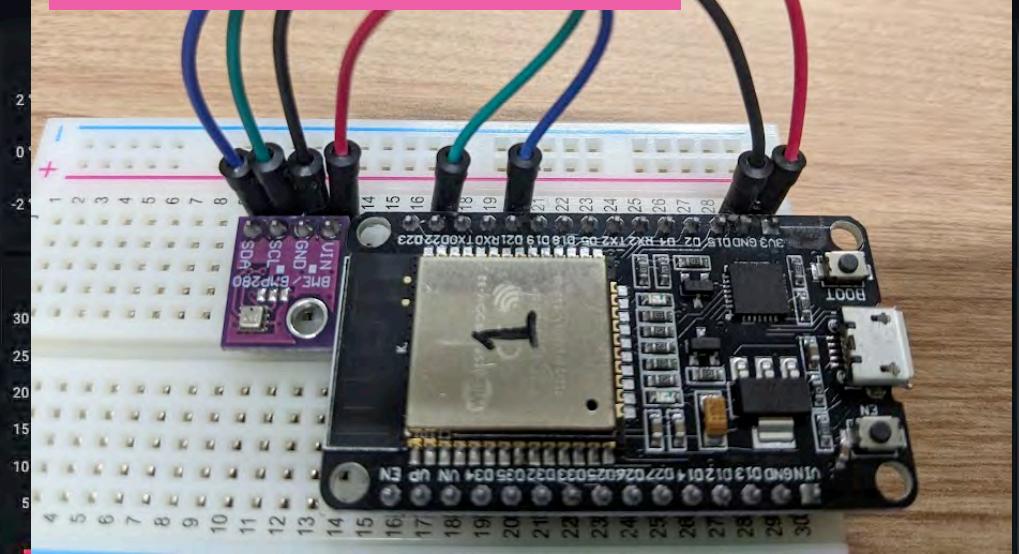
Challenge
10:00 - 11:30
Group : C

B

Final goal : Make your own graph



MCU-sensor wire connection



μPython
Programming
(Copy & paste)
on ESP32

```
from machine import Pin, SoftI2C, unique_id
from time import sleep
import BME280
import socket
import ubinascii
import network
import time

# BME280 setup
SCL = 22
SDA = 21
i2c = SoftI2C(scl=Pin(SCL), sda=Pin(SDA))
bme = BME280.BME280(i2c=i2c)

# Wi-Fi credentials
SSID = "JFWM-WS"
PASSWORD = "goodtime"

print(f"Connecting to Wi-Fi network: {SSID}")
```

Tutorial list

Instruction & sample program

<http://161.34.0.113/>



Preparation

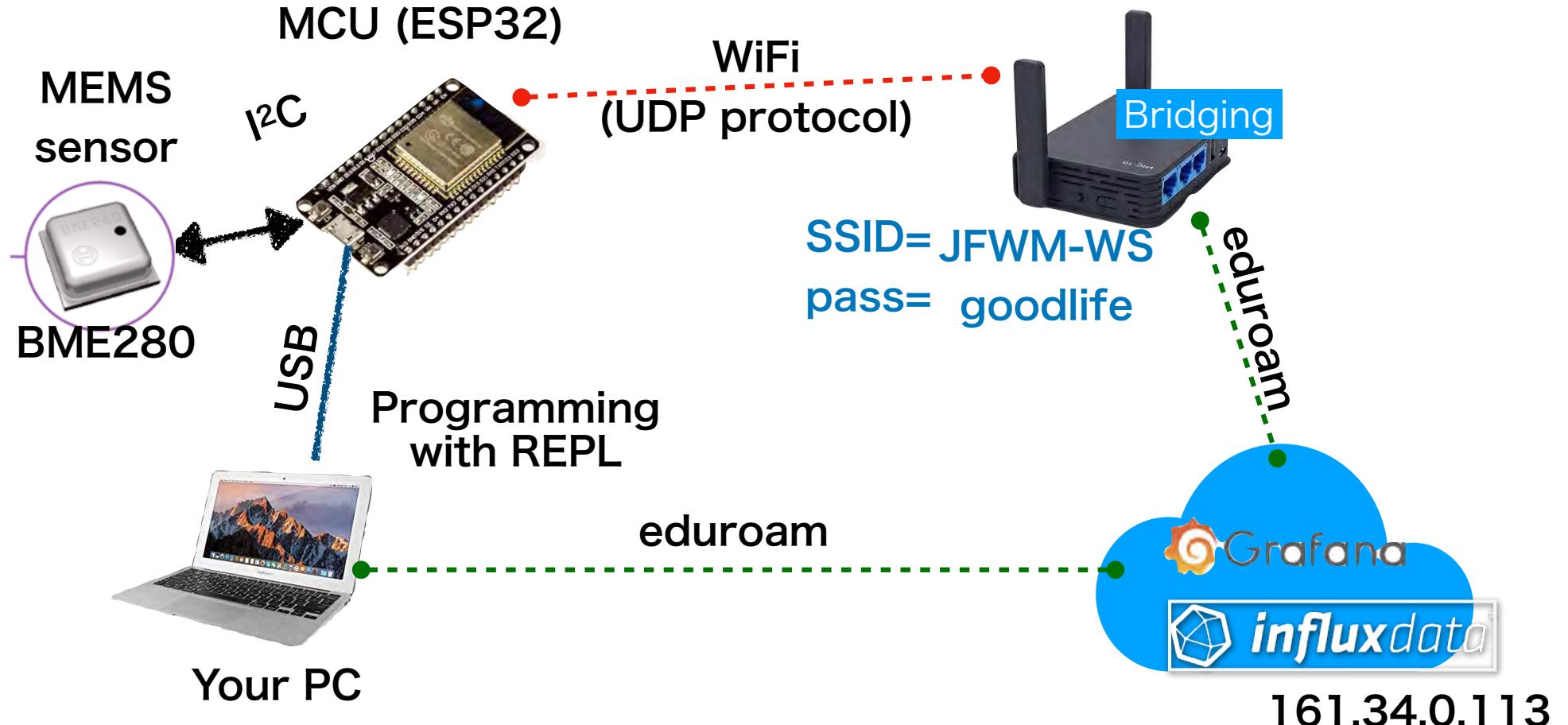
0. MicroPython install on ESP32 by Thonny IDE

Experiments

- 1. Blink LED on ESP32**
- 2. Checking the WiFi function of ESP32**
- 3. Wire BME280 to get data**
(BME280.py library installation)
- 4. Upload BME280 data to cloud via UDP**
- 5. Visualization on Grafana**

(Extra) BLE Beacon RSSI monitoring on Grafana

Overview of connections



Please try connection on your PC

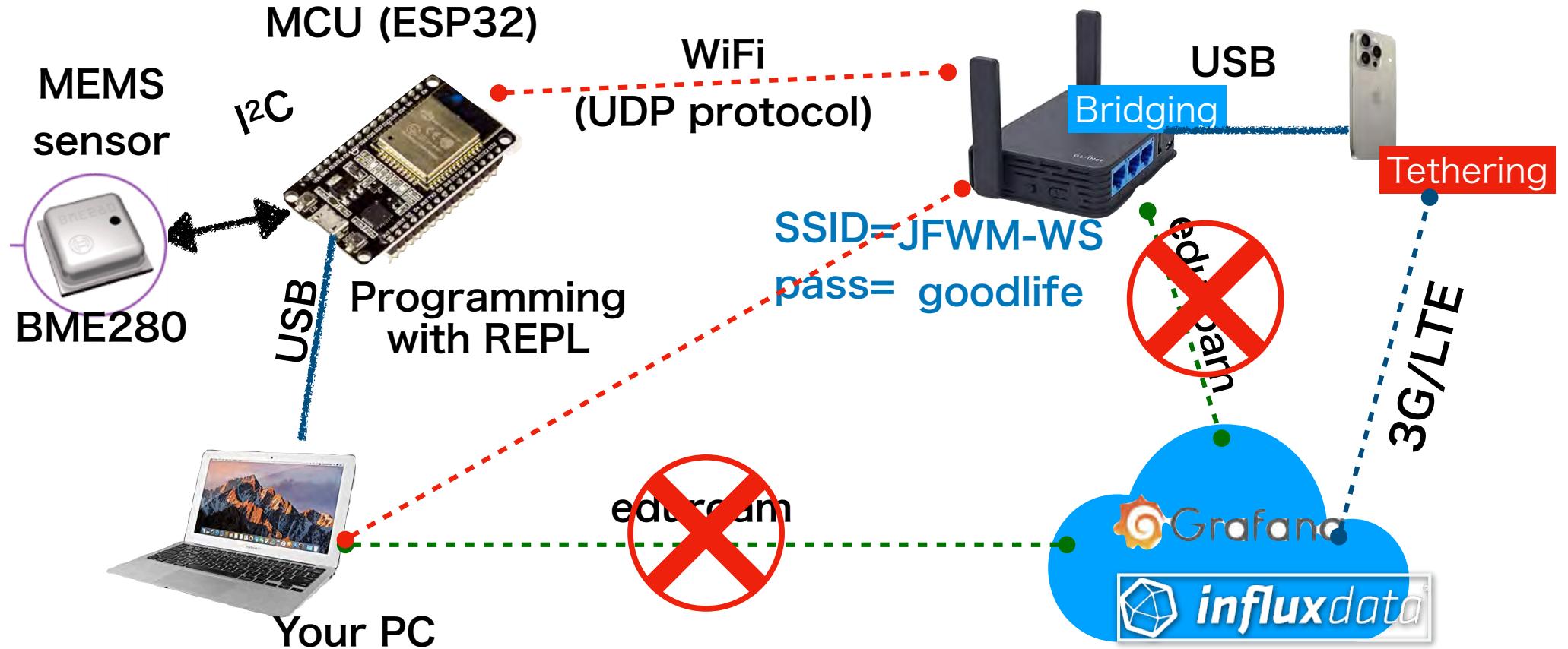
1) Can you to connect <http://161.34.0.113> for instructions ?

Yes

2) Can you to connect <http://161.34.0.113:3000> for Grafana ?

Might be No

Overview of connections (revised)



Note

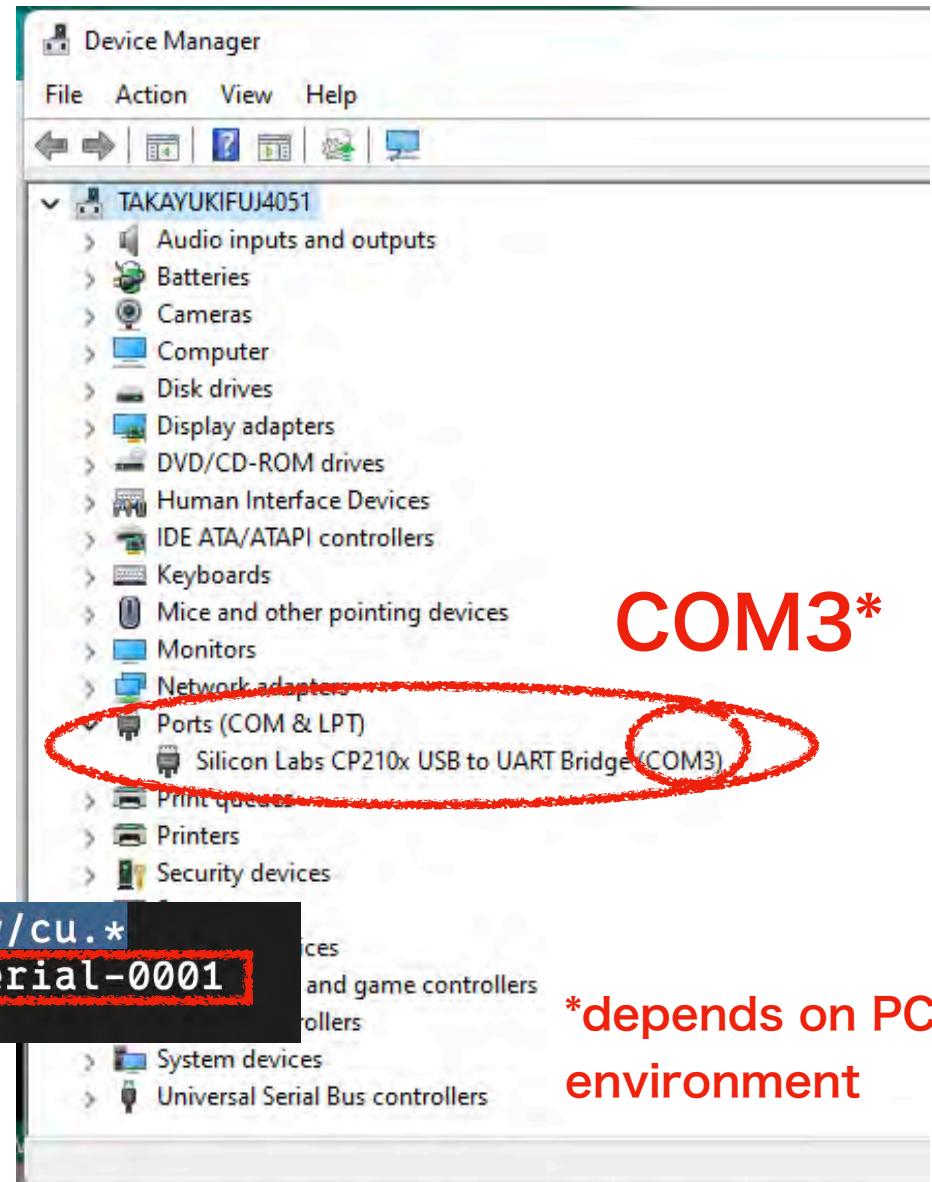
eduroam@USMB's security policy is restricting the Grafana communication

Please connect to '**JFWM-WS**' during the tutorial to connect to Grafana. However, the capacity of tethering is limited, so **only one PC in the group** should be connected, and **do not use large capacity communication such as Youtube, Onedrive.**

Let's connect ESP32 to PC



Confirm **COM port number** on
Device Manager



Connect ESP32 board
to Windows PC /Mac
w/μ-USB cable



On MacOS Terminal App. just input

ls /dev/cu.*

```
(base) takayukifujita@MacBook-Air ~ % ls /dev/cu.*  
/dev/cu.BN0085-BT  
/dev/cu.Bluetooth-Incoming-Port  
/dev/cu.usbserial-0001
```

COM3*

*depends on PC
environment

Tutorial list

Instruction & sample program

<http://161.34.0.113/>



Preparation

MicroPython install on ESP32 by Thonny IDE

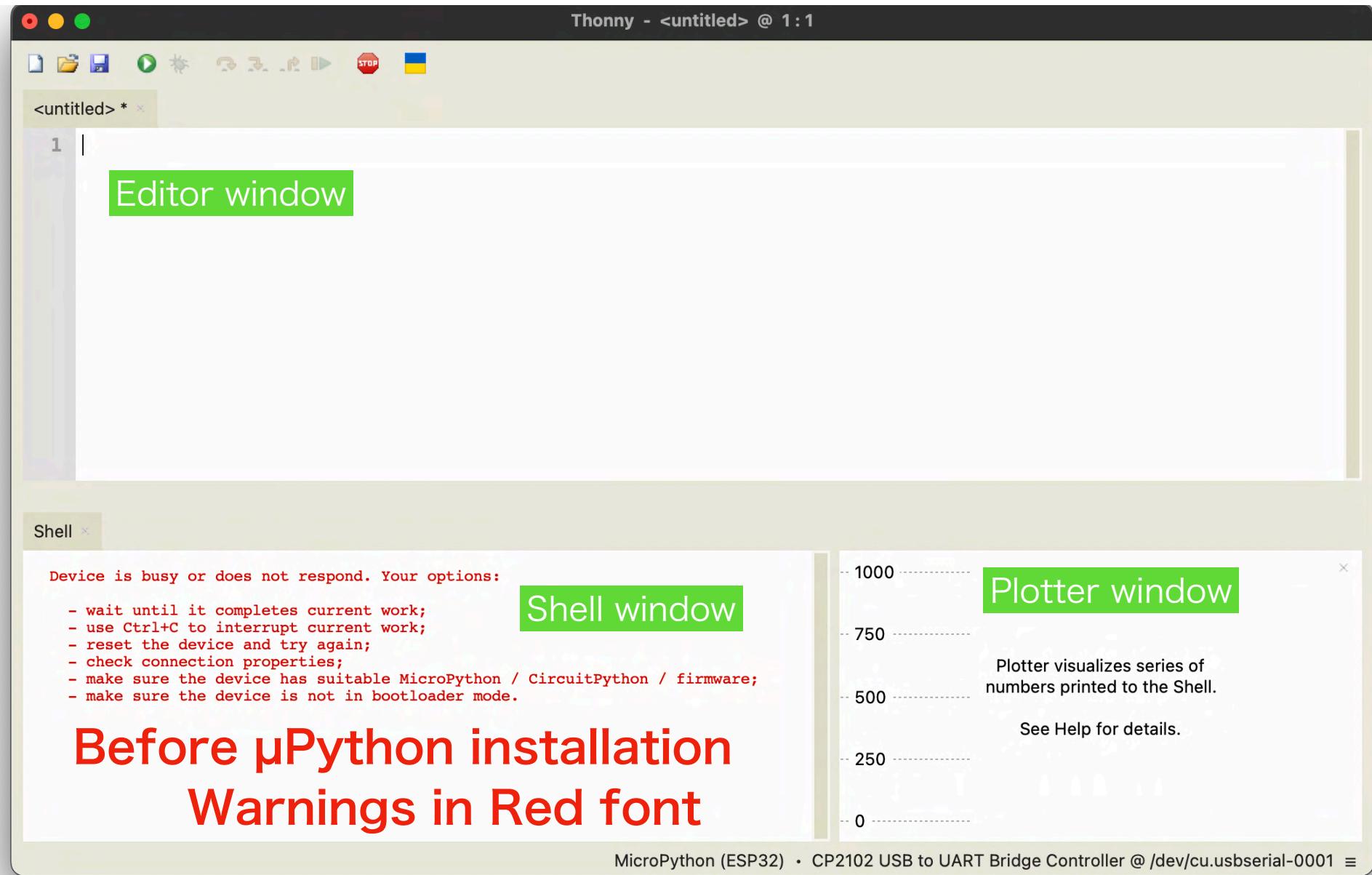
Experiments

1. Blink LED on ESP32
2. Checking the WiFi function of ESP32
3. Wire BME280 to get data
(BME280.py library installation)
4. Upload BME280 data to cloud via UDP
5. Visualization on Grafana

(Extra) BLE Beacon RSSI monitoring on Grafana

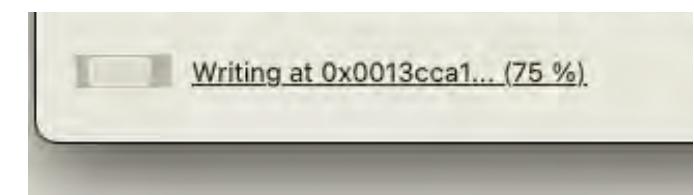
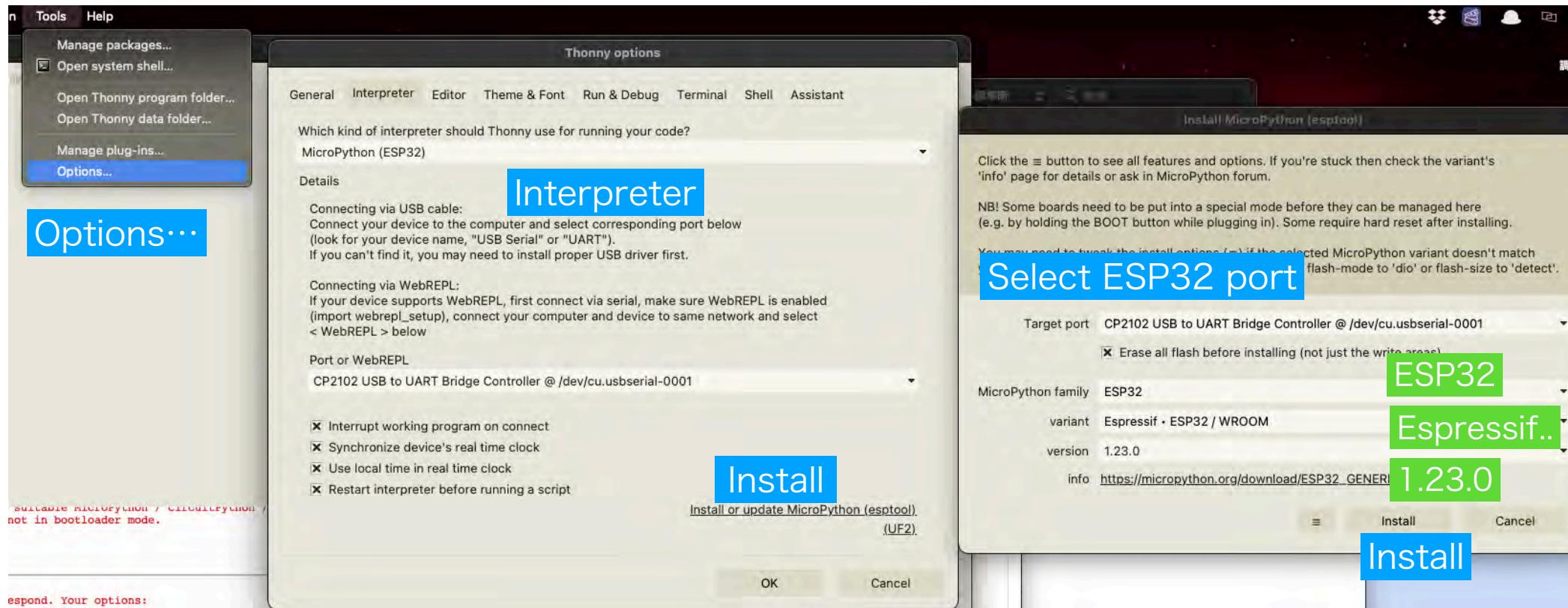
Install μ Python via Thonny app

Sample of Thonny's window



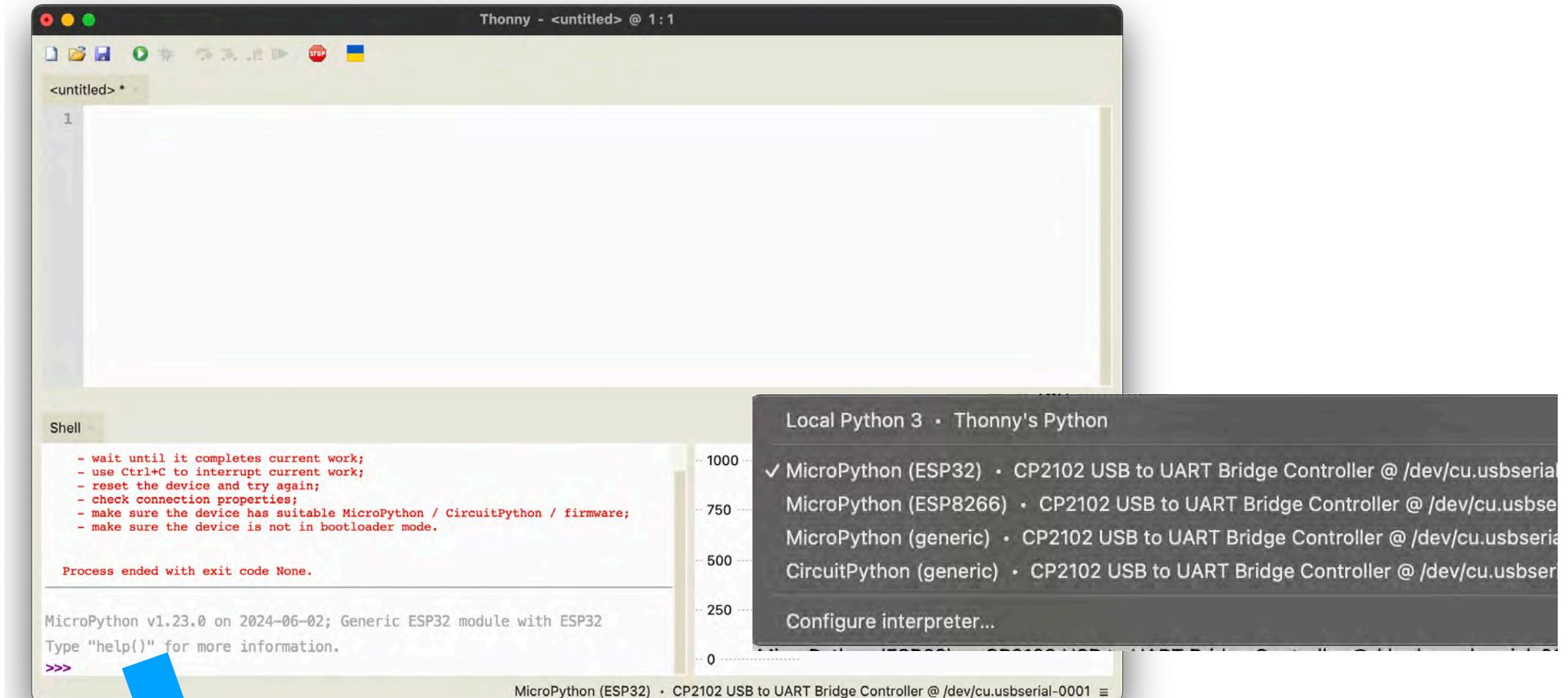
Install μ Python via Thonny app

Download Thonny and install, Run



Wait for writing until "Done"





Click the right bottom to choose interpreter

(ESP32)

Now you can use **μPython** on ESP32

MicroPython v1.23.0 on 2024-06-02; Generic ESP32 module with ESP32

Type "help()" for more information.

>>>

Tutorial list

Instruction & sample program

<http://161.34.0.113/>



Preparation

MicroPython install on ESP32 by Thonny IDE

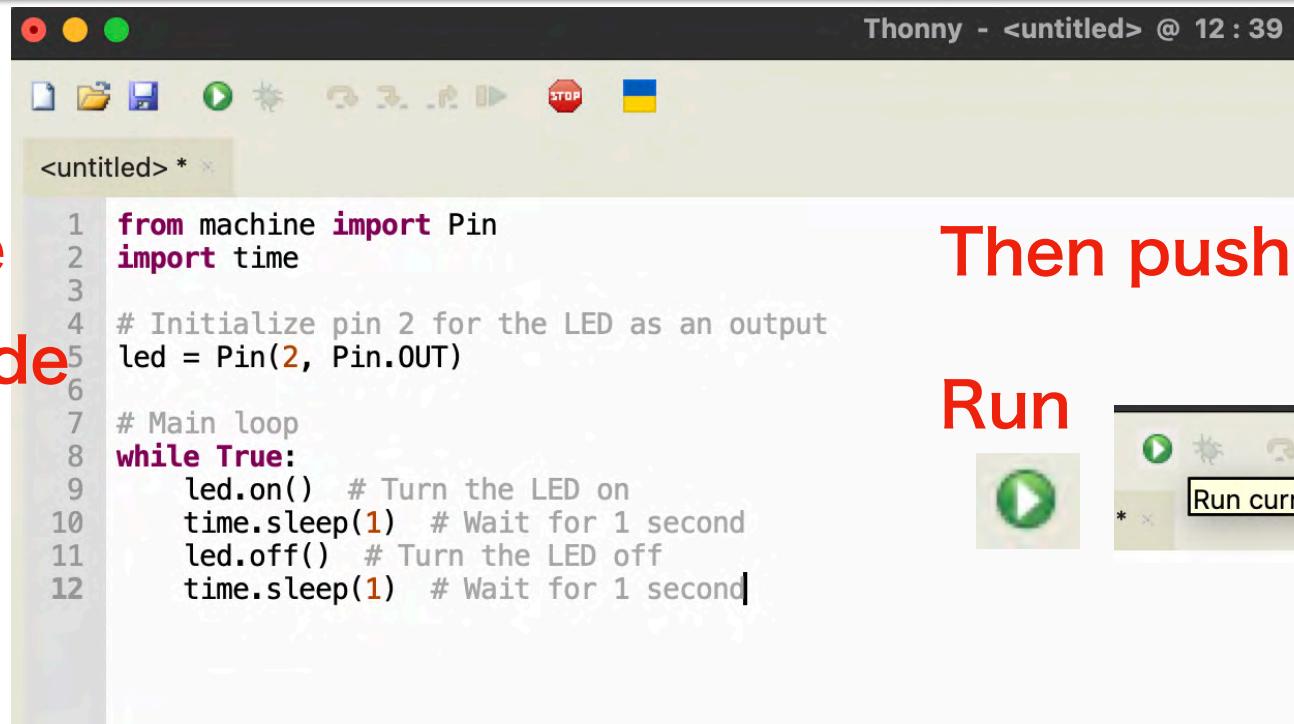
Experiments

1. Blink LED on ESP32
2. Checking the WiFi function of ESP32
3. Wire BME280 to get data
(BME280.py library installation)
4. Upload BME280 data to cloud via UDP
5. Visualization on Grafana

(Extra) BLE Beacon RSSI monitoring on Grafana

Ex1) Blink LED

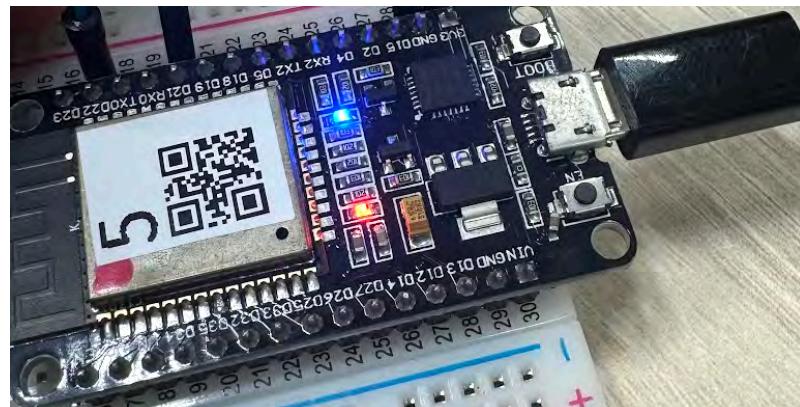
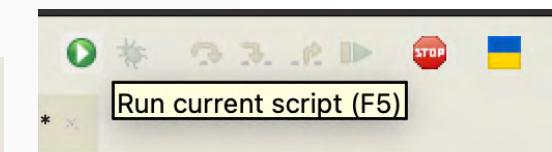
Copy & paste
Python code



The screenshot shows the Thonny IDE interface with the title bar "Thonny - <untitled> @ 12 : 39". The toolbar includes icons for file operations, run, stop, and a UK flag. A script window titled "<untitled> *" contains the following Python code:

```
1 from machine import Pin
2 import time
3
4 # Initialize pin 2 for the LED as an output
5 led = Pin(2, Pin.OUT)
6
7 # Main loop
8 while True:
9     led.on() # Turn the LED on
10    time.sleep(1) # Wait for 1 second
11    led.off() # Turn the LED off
12    time.sleep(1) # Wait for 1 second
```

Then push
Run



Blue LED(pin 2) blinking

Try to change blink intervals

Just change time.sleep and Run



Tutorial list

Instruction & sample program

<http://161.34.0.113/>



Preparation

MicroPython install on ESP32 by Thonny IDE

Experiments

1. Blink LED on ESP32
2. Checking the WiFi function of ESP32
3. Wire BME280 to get data
(BME280.py library installation)
4. Upload BME280 data to cloud via UDP
5. Visualization on Grafana

(Extra) BLE Beacon RSSI monitoring on Grafana

Ex2) Wi-Fi finder nearby

Copy & paste
Python code

Then push

Run 

```
import network
import time

# Initialize the WiFi interface in station mode
wlan = network.WLAN(network.STA_IF)

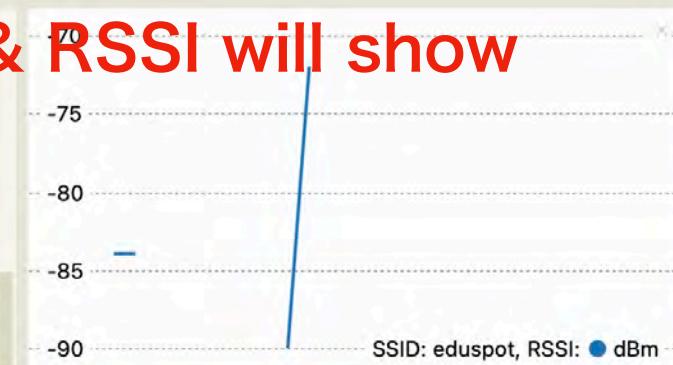
# Activate WiFi
wlan.active(True)

print("Nearby WiFi networks:")
for _ in range(5): # Repeat the scan 5 times
    networks = wlan.scan()
    for ssid, bssid, channel, rssi, authmode, hidden in networks:
        print(f"SSID: {ssid.decode()}, RSSI: {rssi} dBm")
    time.sleep(1) # Wait for 1 second
```

Shell

```
SSID: eduroam, RSSI: -72 dBm
SSID: , RSSI: -72 dBm
SSID: eduspot, RSSI: -78 dBm
SSID: eduroam, RSSI: -79 dBm
SSID: , RSSI: -79 dBm
SSID: eduspot, RSSI: -82 dBm
SSID: eduroam, RSSI: -82 dBm
SSID: , RSSI: -82 dBm
SSID: eduroam, RSSI: -84 dBm
SSID: eduspot, RSSI: -86 dBm
SSID: Alice Village by CA, RSSI: -87 dBm
SSID: showroom, RSSI: -89 dBm
SSID: eduspot, RSSI: -89 dBm
SSID: eduroam, RSSI: -89 dBm
SSID: , RSSI: -89 dBm
SSID: eduspot, RSSI: -90 dBm
```

Nearby SSID & RSSI will show



MicroPython (ESP32) • CP2102 USB to UART Bridge Controller @ /dev/cu.usbserial-0001 ≡

Finding SSID “JFWM-WS”

Tutorial list

Instruction & sample program

<http://161.34.0.113/>



Preparation

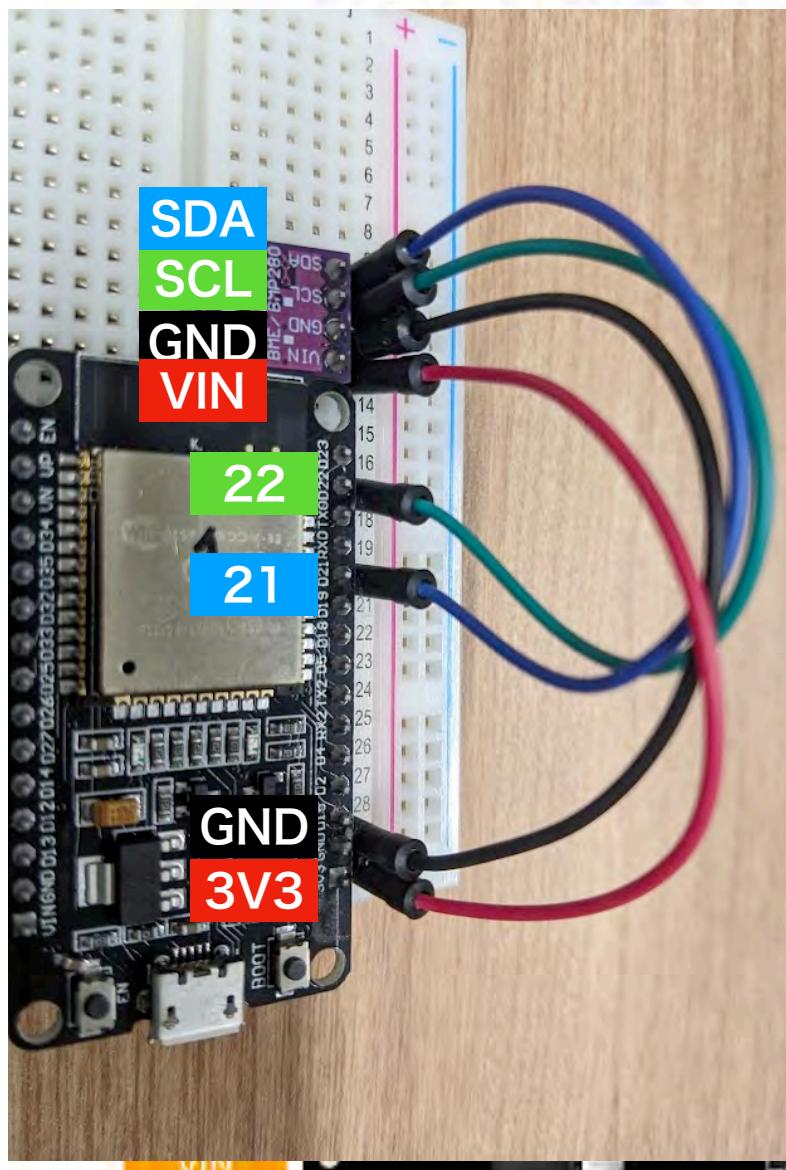
MicroPython install on ESP32 by Thonny IDE

Experiments

1. Blink LED on ESP32
2. Checking the WiFi function of ESP32
3. Wire BME280 to get data
(BME280.py library installation)
4. Upload BME280 data to cloud via UDP
5. Visualization on Grafana

(Extra) BLE Beacon RSSI monitoring on Grafana

Wire connection on breadboard



4 wires for I2C connection

Serial clock

Serial data

Power line



Ex3) BME280 data get

Copy & paste

Python code

Thonny - <untitled> @ 17 : 13

```
from time import sleep
import BME280

# BME280 setup
SCL = 22
SDA = 21
i2c = SoftI2C(scl=Pin(SCL), sda=Pin(SDA))
bme = BME280.BME280(i2c=i2c)

while True:
    temp = bme.temperature
    hum = bme.humidity
    press = bme.pressure
    print(f"Temperature: {temp}°C, Humidity: {hum}%, Pressure: {press}hPa")
    sleep(1)
```

Then push Run 

You will get error message

```
>>> %Run -c $EDITOR_CONTENT
MPY: soft reboot
Traceback (most recent call last):
  File "<stdin>", line 3, in <module>
ImportError: no module named 'BME280'
```

no module named 'BME280'

Need to install 'BME280.py' library

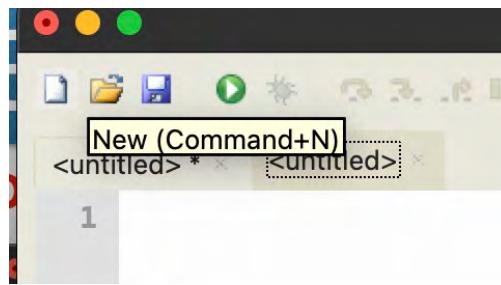
Plotter visualizes series of numbers printed to the Shell.

See Help for details.

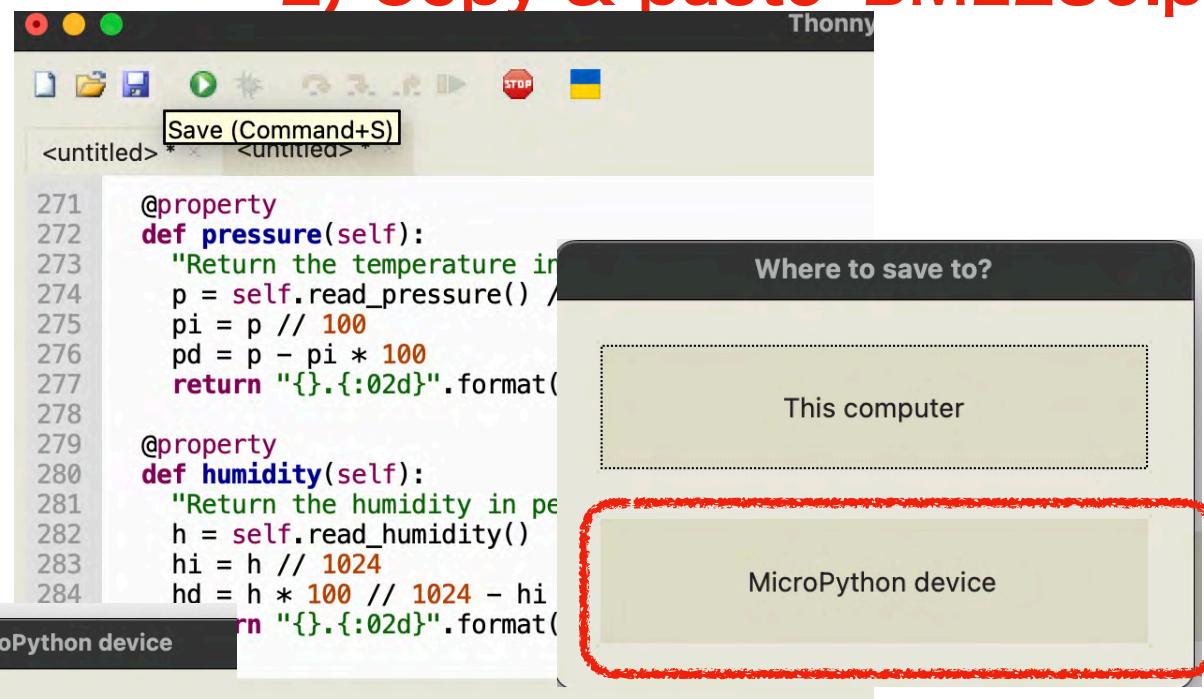
MicroPython (ESP32) • CP2102 USB to UART Bridge Controller @ /dev/cu.usbserial-0001 ≡

Ex3) BME280 data get

1) Create new editor window



2) Copy & paste 'BME280.py'



3) Then Save it to ESP32



4) Set file name 'BME280.py'

Save > [OK]

Ex3) BME280 data get

Press initial tab (Untitled..)

Then push

Run 

```
from time import sleep
import BME280

# BME280 setup
SCL = 22
SDA = 21
i2c = SoftI2C(scl=Pin(SCL), sda=Pin(SDA))
bme = BME280.BME280(i2c=i2c)

while True:
    temp = bme.temperature
    hum = bme.humidity
    press = bme.pressure
    print(f"Temperature: {temp}°C, Humidity: {hum}%, Pressure: {press}hPa")
    sleep(1)
```

Shell

```
>>> %Run -c $EDITOR_CONTENT
```

```
MPY: soft reboot
Temperature: 22.13°C, Humidity: 83.26%, Pressure: 691.48hPa
Temperature: 24.99°C, Humidity: 32.04%, Pressure: 953.16hPa
Temperature: 24.98°C, Humidity: 32.03%, Pressure: 953.11hPa
Temperature: 24.98°C, Humidity: 32.03%, Pressure: 953.19hPa
Temperature: 24.98°C, Humidity: 32.02%, Pressure: 953.19hPa
Temperature: 24.98°C, Humidity: 32.01%, Pressure: 953.14hPa
Temperature: 24.99°C, Humidity: 32.02%, Pressure: 953.14hPa
```

You will get data from BME280 sensor

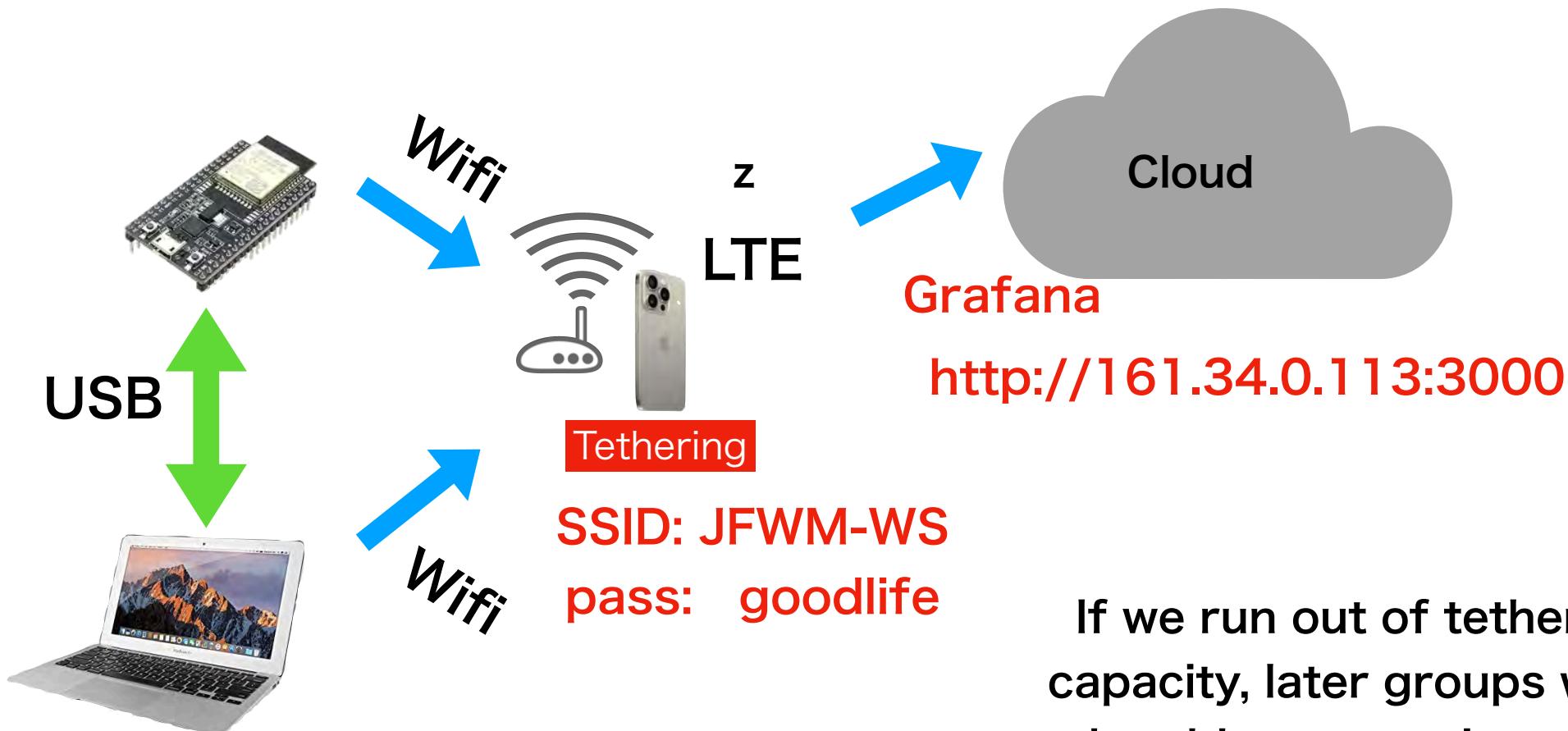
-500 Temperature: ● °C, Humidity: ● %, Pressure: ● .14hPa

MicroPython (ESP32) · CP2102 USB to UART Bridge Controller @ /dev/cu.usbserial-0001 ≡

CAUTION

Due to the limited capacity of tethering,
please connect to the following AP and
setup Grafana only during the tutorial

Never use Youtube, cloud storage, or other large traffic services



One PC connection for each group

If we run out of tethering capacity, later groups won't be able to experiment & Fujita's trip in France will be disastrous.

Ex4) BME280 data upload to InfluxDB

Copy & paste

Then push

The screenshot shows the Thonny IDE interface. The top bar displays "Thonny - <untitled> @ 42 : 19". Below the bar, there are two tabs: "<untitled> *" and "[BME280.py] *". The "[BME280.py]" tab contains the following Python code:

```
38 sock = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
39
40 def send_to_influxdb(temp, hum, press):
41     measurement = f"ble_rssi_{mac_address}"
42     data = f"{measurement} temp={temp},hum={hum},press={press}"
43     sock.sendto(data.encode(), (INFLUXDB_IP, INFLUXDB_PORT))
44
45 while True:
46     temp = bme.temperature
47     hum = bme.humidity
48     press = bme.pressure
49     print(f"Temperature: {temp}°C, Humidity: {hum}%, Pressure: {press}hPa")
50
51     send_to_influxdb(temp, hum, press)
52
53     sleep(1)
```

The code uses a socket to send data to an InfluxDB instance at 161.34.0.113. It prints the current temperature, humidity, and pressure to the shell. The "Run" button is visible in the toolbar.

Large red text overlays are present in the center of the screen:

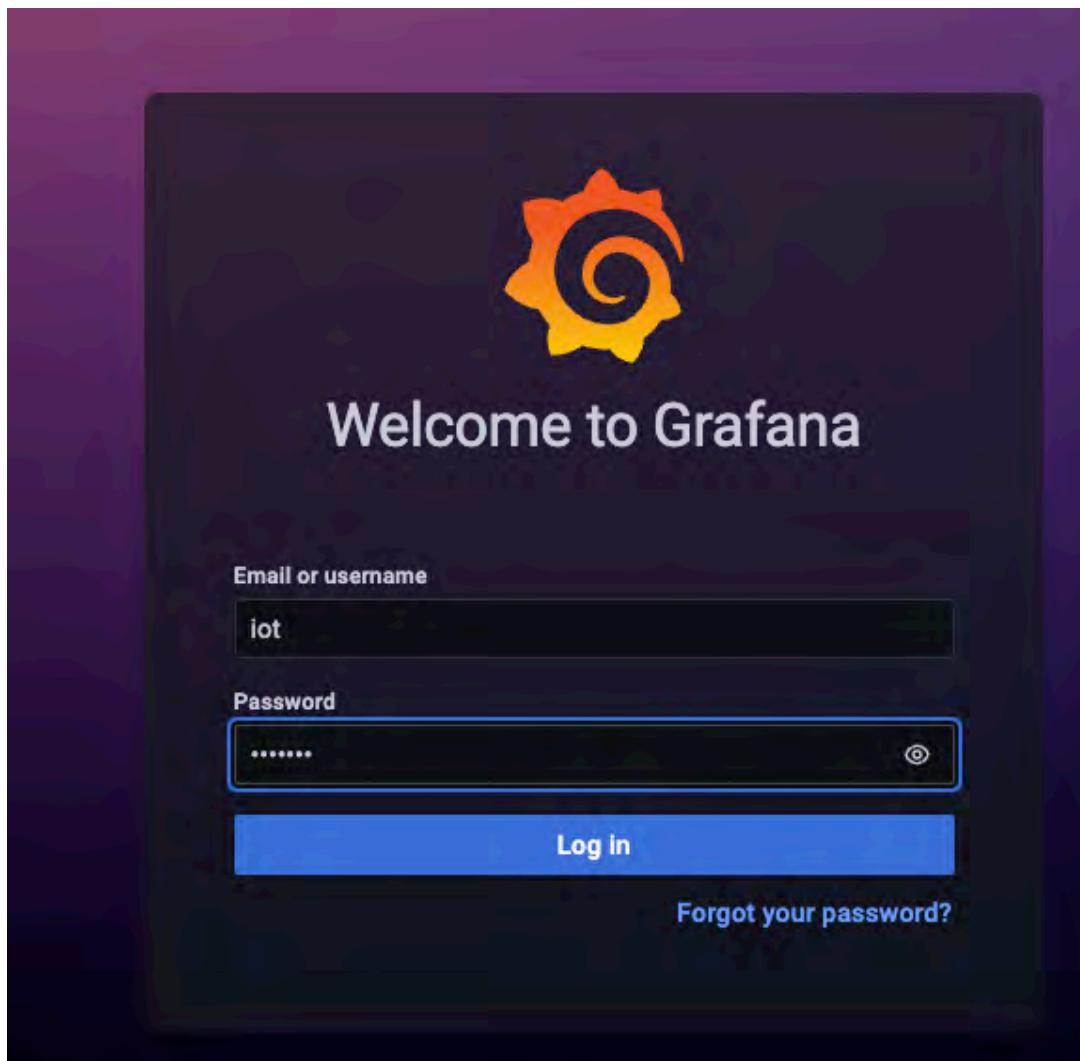
- Temperature, Humidity, and Pressure
- Data will be uploaded to InfluxDB on Cloud (161.34.0.113)
- Via Wifi-LTE(Tethering) connection
- SSID: JFWM-WS, Pass: goodlife

The bottom status bar shows "MicroPython (ESP32) • CP2102 USB to UART Bridge Controller @ /dev/cu.usbserial-0001".

Data visualization on Grafana

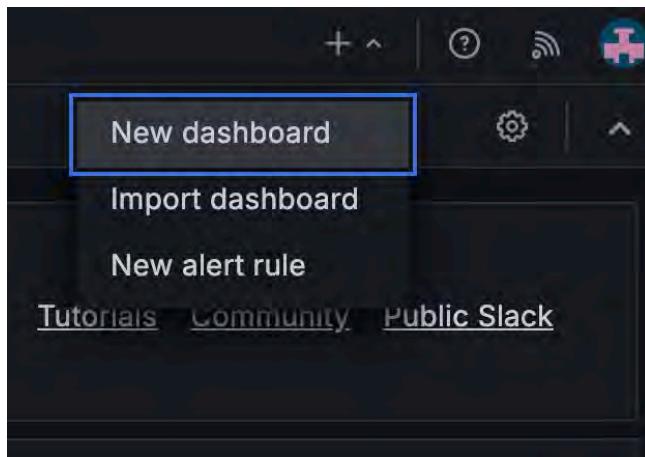
Access grafana server on cloud via AP
(SSID: JFWM-WS, pass: goodlife)

<http://161.34.0.113:3000/>

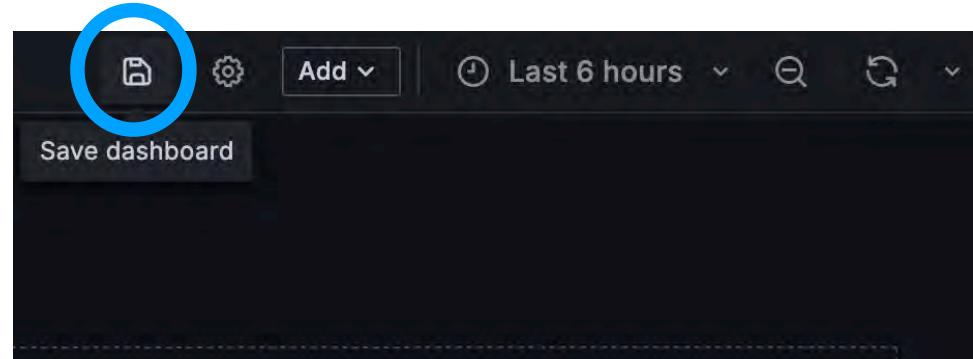


user: **iomt**
pass: **iomt2023**

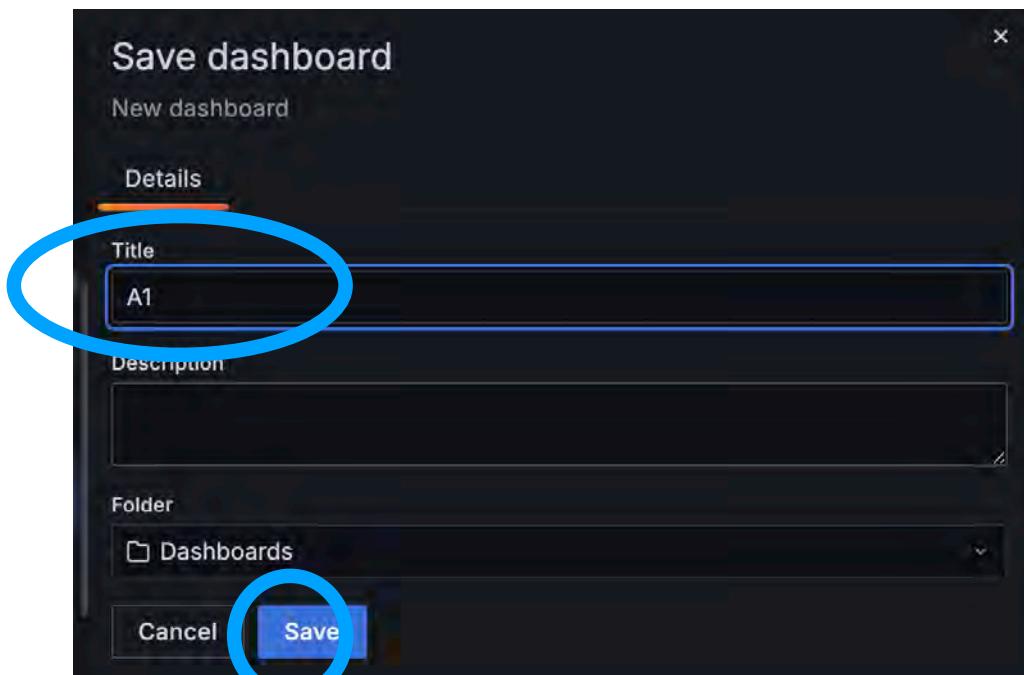
(1) create “Dashboard”



(2) Save...

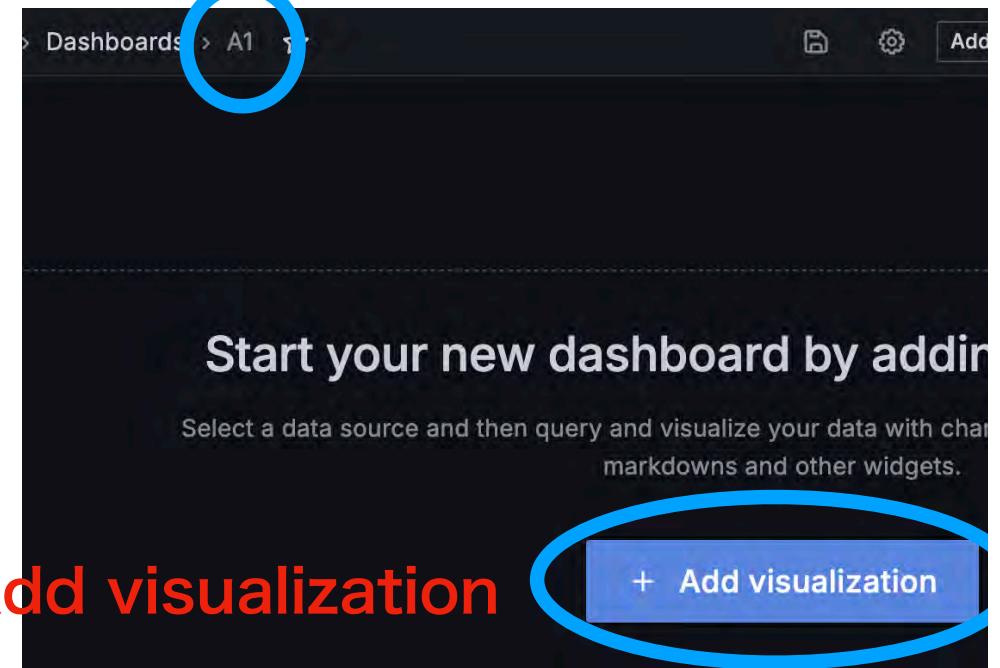


(3) Input title as your group A1…C3



(4) Save

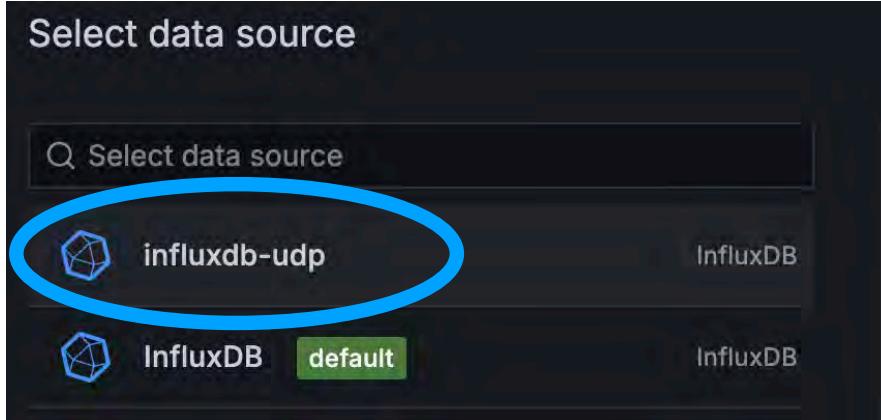
Confirm your group name



(5) Add visualization

(6) Select data source

Data from ESP32 is stored
in influxdb-udp



(7) Press 'influxdb-udp'
> Edit panel appears

The screenshot shows the Grafana dashboard editor. The top navigation bar includes 'Home', 'Dashboards', 'A1', and 'Edit panel'. On the left, there's a warning message 'Panel Title' with an exclamation mark icon. The main area displays 'No data'. The bottom navigation bar has tabs for 'Query' (which is active, highlighted in orange), 'Transform data', 'Alert', and 'Panel'. The 'Data source' dropdown is set to 'influxdb-udp'. The 'Query' editor shows a single measurement 'A' from the 'influxdb-udp' data source. The query itself is: `SELECT field(value) X mean() X`.

Query 1

Transform data 0

Alert 0

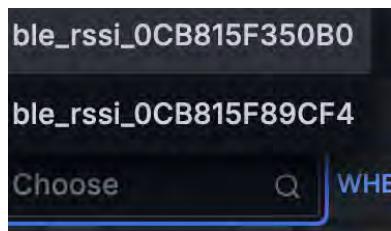
Data source influxdb-udp

Query ... MD = auto = 814 Interval = 30s Query inspector

FROM default select measurement WHERE +

SELECT field(value) × mean() × +

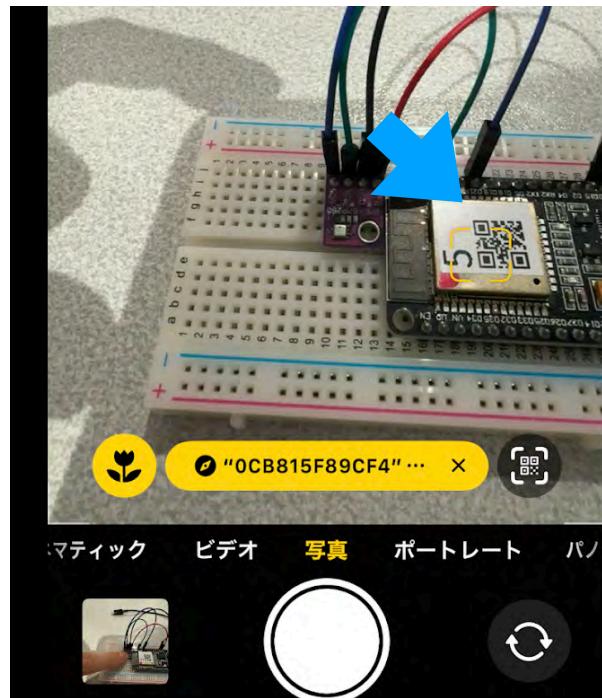
(8) Select measurement



Finding ble_rssi_?????????

????????? is MAC address
of your ESP32

You can get address by
QR code reader



Select your group's address one

Or labelled number on
ESP32 and address

1. 0CB815F89518
2. 0CB815F58564
3. 0CB815F72A38
4. 0CB815F350B0
5. 0CB815F89CF4
6. 0CB815F66E08

Query 1 Transform data 0 Alert 0

Data source influxdb-udp ? Query ... MD = auto = 814 Interval = 30s Query inspector

A (influxdb-udp)

FROM default select measurement WHERE +

SELECT field(value) × mean() × +

(9) Select field (value)

Transform data 0 Alert 0

default ble_rssi_0CB815F350B0 × W

field Choose X me

time(hum) X +

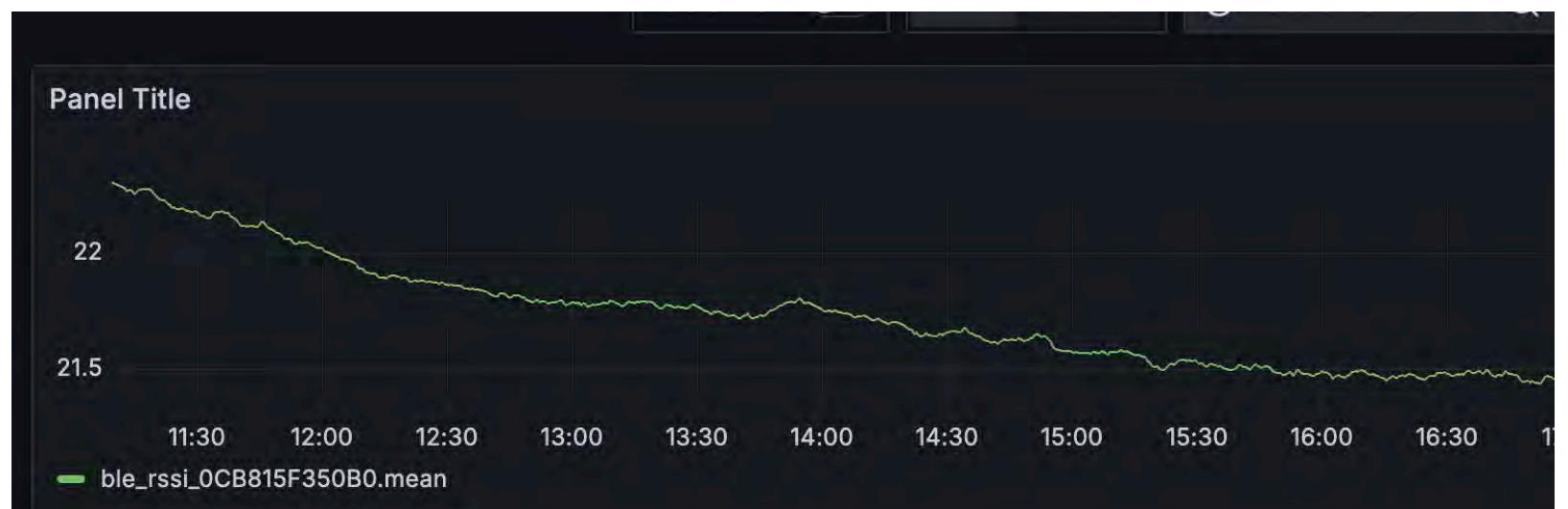
(opti press TIME asc)

(opti temp (optional))

You can choose
hum: Humidity
press: Pressure
temp: Temperature

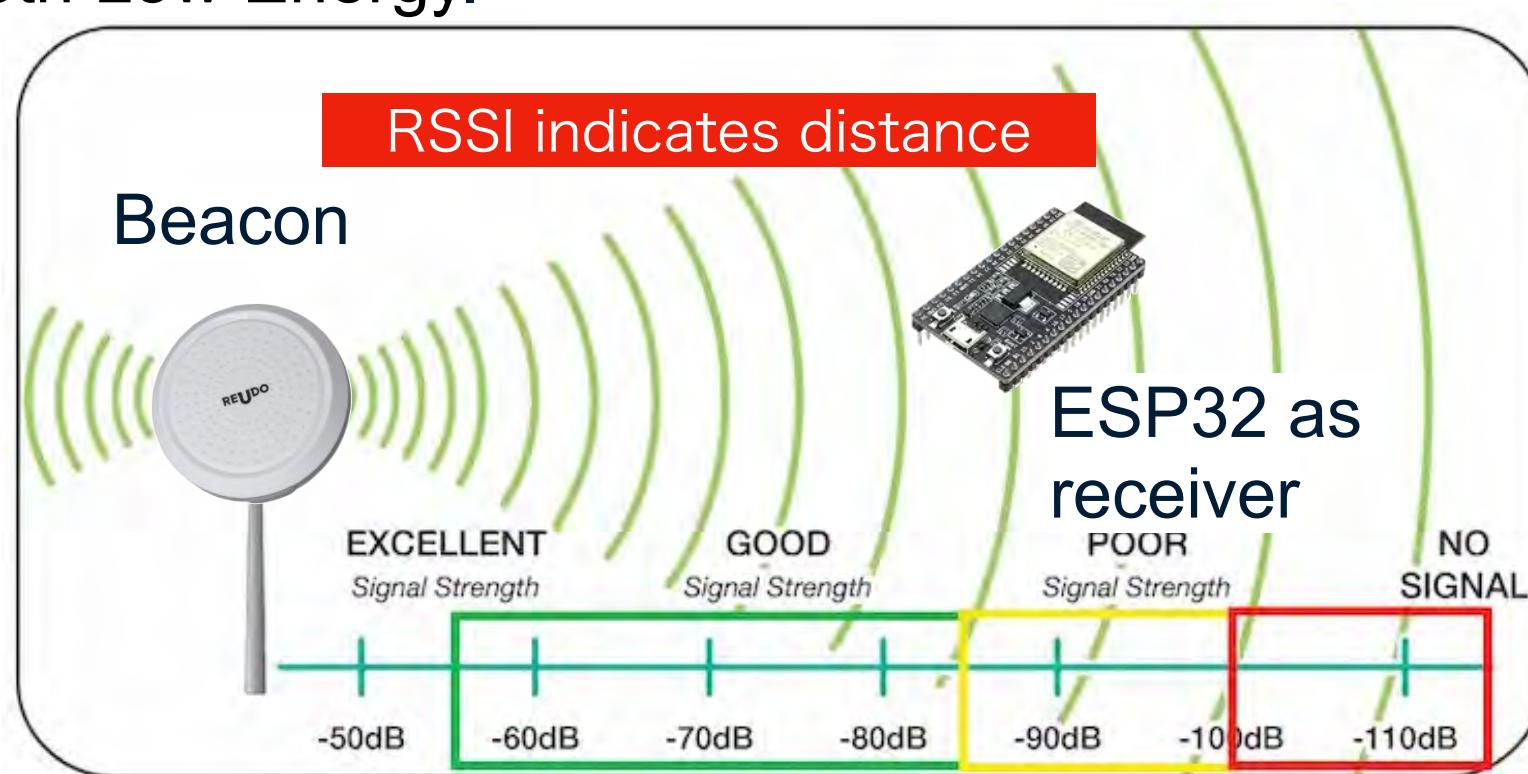
Details will
be explained
on site

(10)
Finally, get a graph
for selection



Opt) Beacon RSSI upload InfluxDB

Bluetooth Low Energy (BLE) beacons are small, battery-powered devices that transmit data using Bluetooth Low Energy.



Before programing
Power on Beacon



Press and hold the button



LED flashes
and SW ON

Opt) Beacon RSSI upload InfluxDB

Copy & paste

Then push

Thonny - <untitled> @ 17:6

<untitled> * [BME280.py] *

```
8 SSID = "Duplex_loft"
9 PASSWORD = "DUPLEX_LOFT"
10
11 # InfluxDB server details
12 INFLUXDB_IP = "161.34.0.113"
13 INFLUXDB_PORT = 8089
14
15 # Target MAC addresses (add or remove as needed)
16 TARGET_MACS = [
17     "ce3f413259a9", # Fujita's Mi-band
18 ]
19 print("Processing target MAC addresses...")
20 target_macs = [mac.replace(':', '').lower() for mac in TARGET_MACS]
21 print(f"Loaded {len(target_macs)} target MAC addresses")
22
23 print("Initializing BLE...")
24 ble = BLE()
25 ble.active(True)
26 print("BLE initialized")
27
28 print(f"Connecting to Wi-Fi network: {SSID}")
29
```

Shell

```
Target device found. MAC: ce3f413259a9, RSSI: -70
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-51,ce3f413259a9=-70
Target device found. MAC: ce3f413259a9, RSSI: -76
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-49,ce3f413259a9=-76
Target device found. MAC: ce3f413259a9, RSSI: -71
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-52,ce3f413259a9=-71
Target device found. MAC: ce3f413259a9, RSSI: -70
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-51,ce3f413259a9=-70
Target device found. MAC: ce3f413259a9, RSSI: -69
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-51,ce3f413259a9=-69
Target device found. MAC: ce3f413259a9, RSSI: -71
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-50,ce3f413259a9=-71
Target device found. MAC: ce3f413259a9, RSSI: -70
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-52,ce3f413259a9=-70
Target device found. MAC: ce3f413259a9, RSSI: -71
Sent to InfluxDB: ble_rssi_0CB815F89CF4 Uplink_RSSI=-52,ce3f413259a9=-71
```

Run 

Modify the TARGET address to your Beacon address



Read QR code

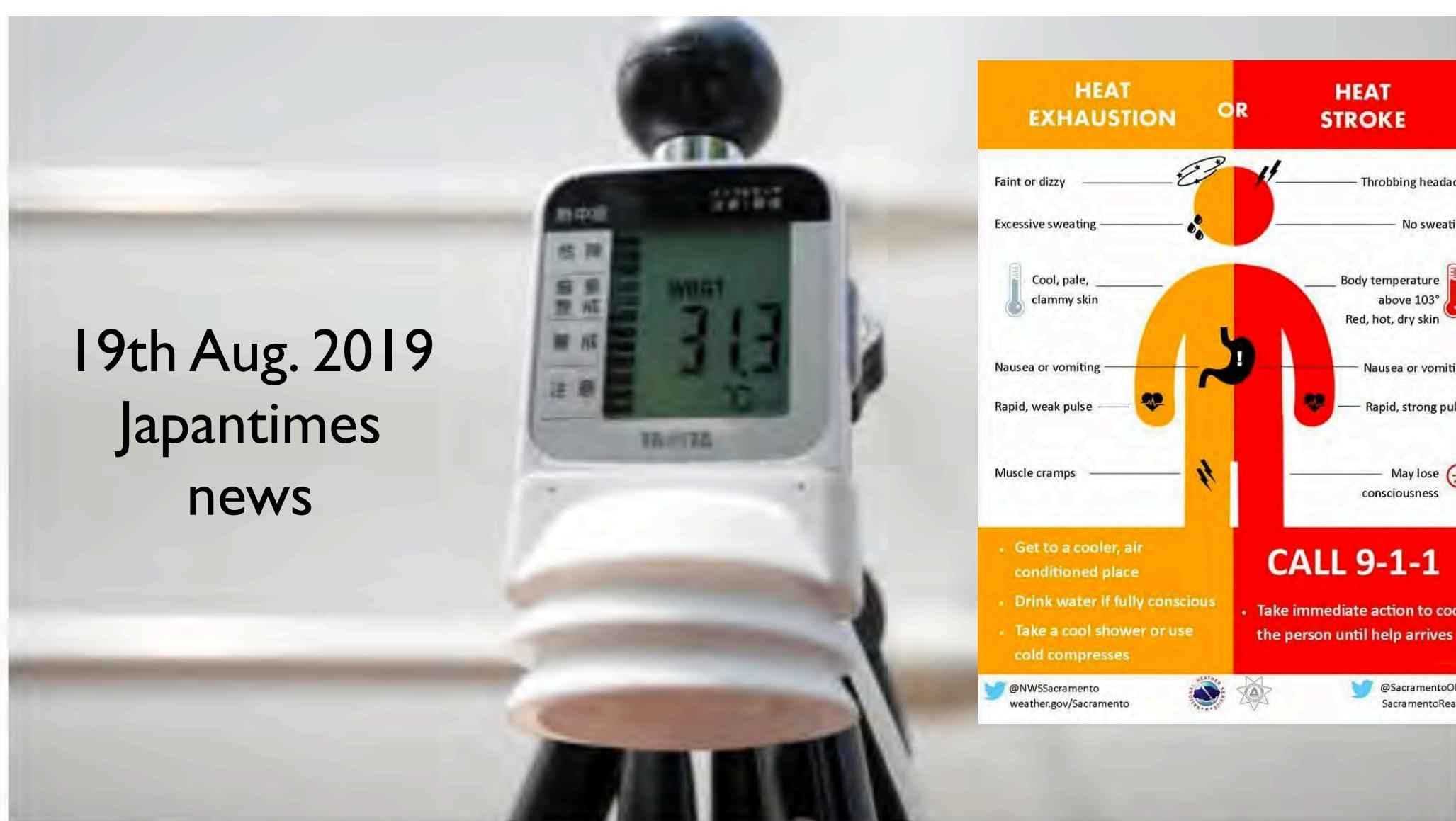


Visualize on Grafana

RSSI values for specific Beacon address and uplink to AP will be uploaded

If you still have time and energy
left over, you can modify a
program to upload the **WBGT**
values from **BME280** sensor

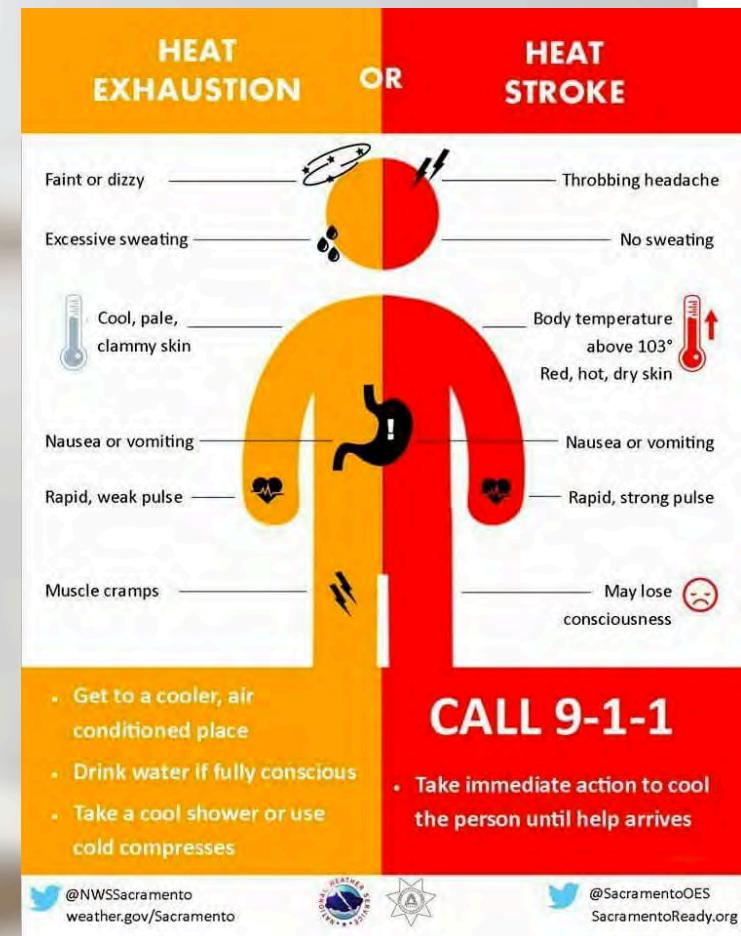
19th Aug. 2019
Japantimes
news



A temperature indicator measures 31.3 degrees Celsius during a test of heat countermeasures for the Tokyo 2020 Olympic and Paralympic Games in the Japanese capital on July 25. | REUTERS

NATIONAL

Heatstroke death toll since July exceeds 100 in Tokyo's 23 wards



@NWSSacramento
weather.gov/Sacramento



@SacramentoOES
SacramentoReady.org

Wet Bulb Globe Temperature (WBGT) from Temperature and Relative Humidity

Temperature in Degrees Celsius

	20	22	24	26	28	30	32	34	36	38	
0	14.8	16.1	18.0	18.6	19.8	21.1	22.3	23.5	24.7	25.8	
5	15.3	16.7	18.7	19.4	20.7	22.0	23.3	24.6	25.9	27.2	
10	16.0	17.4	19.4	20.2	21.6	23.0	24.3	25.7	27.1	28.4	
15	16.5	18.0	20.1	20.9	22.4	23.8	25.2	26.7	28.1	29.5	
20	17.1	18.7	20.8	21.6	23.1	24.6	26.2	27.7	29.2	30.5	
25	17.6	19.3	21.4	22.3	24.0	25.5	27.0	28.6	30.1	31.7	33.2
30	18.2	19.8	22.0	23.0	24.6	26.2	27.8	29.4	31.0	32.7	34.2
35	18.7	20.3	22.6	23.6	25.3	26.9	28.6	30.2	31.9	33.5	35.2
40	19.3	20.9	23.2	24.3	26.0	27.6	29.4	31.0	32.7	34.4	36.1
45	19.7	21.5	23.8	24.9	26.6	28.3	30.1	31.8	33.5	35.2	37.0
50	20.2	22.0	24.3	25.5	27.3	29.0	30.8	32.5			
55	20.7	22.4	24.8	26.0	27.8	29.6	31.4	33.3			
60	21.1	22.9	25.4	26.6	28.4	30.2	32.1	34.0			
65	21.6	23.2	25.9	27.1	29.0	30.9	32.7	34.5			
70	22.1	23.9	26.4	27.6	29.4	31.4	33.3	35.1			
75	22.5	24.4	26.9	28.2	30.1	32.0	33.8	35.8			
80	22.9	24.8	27.4	28.7	30.6	32.5	34.4	36.3			
85	23.3	25.2	27.8	29.2	31.1	33.0	35.0	36.9			
90	23.7	25.7	28.3	29.6	31.6	33.5	35.5	37.5			
95	24.2	26.1	28.7	30.1	32.0	34.0	36.0	38.0			
100	24.5	26.5	29.1	30.5	32.5	34.5	36.5	38.5			

Sahara Desert



VHigh temp.
Low Hum

Calculate
by temp.
and humidity

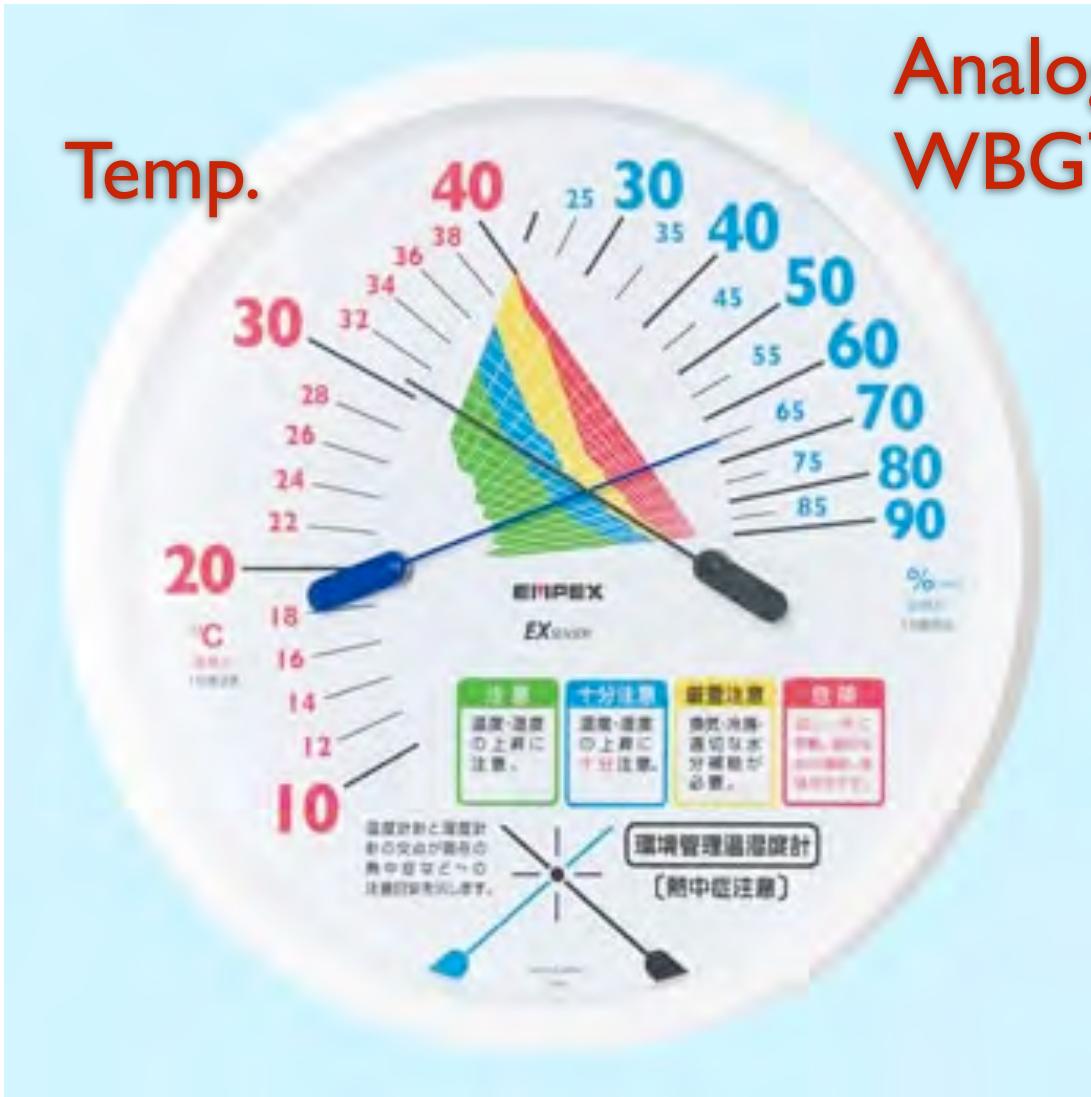
Kyoto



High temp.
High Hum

NOTE: This chart is calculated using temperature and humidity, assuming a very clear sky (maximal solar load), and atmospheric pressure of 1ATA (760 mmHg). Chart A was developed by Professor Yoram Epstein to be used in Ariel's Checklist for hikers in Israel.

Heatstroke WBGT meter



Analog type
WBGT meter

Cross point
= WBGT
Heatstroke



Requires special equipment

$$\text{WBGT} = 0.7T_w + 0.2T_g + 0.1T_d$$

where

- T_w = Natural wet-bulb temperature (combined with dry-bulb temperature indicates humidity)
- T_g = Globe thermometer temperature (measured with a globe thermometer, also known as a black globe thermometer)
- T_d = Dry-bulb temperature (actual air temperature)
- Temperatures may be in either Celsius or Fahrenheit

Indoors, or when solar radiation is negligible, the following formula is often used:

$$\text{WBGT} = 0.7T_w + 0.3T_g$$

Difficult to get
Wet-bulb temp.



A U.S. Navy technician checking the wet-bulb globe temperature at the [Corry Station Naval Technical Training Center](#), Florida

Three sensors
Required for outdoor

How to get WBGT
from °C + %Rh.



Approximation equation

$WBGT =$

$$(Humidity-20) \cdot ((Temperature-40)^2 \cdot (-0.00025) + 0.185) + \frac{11}{15} \cdot (Temperature-25) + 17.8$$

For program

```
WBGT = (hum -20.0) * ( (-0.00025) *  
pow ((temp - 40.0 ),2) + 0.185) + (11 /  
15) * (temp - 25.0) + 17.8;
```