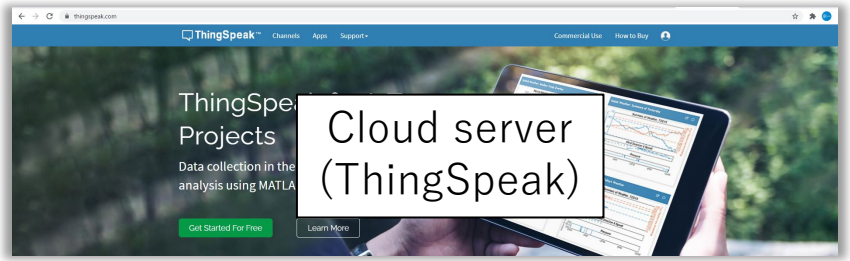
IOTを用いた実験状況の監視システム
Remote monitoring by IOT

Munekazu Horikoshi

Osaka City University

第3回 冷却原子研究会「アトムの会」 2021年8月10日（火）-11日（水）、オンライン開催

Today's goal

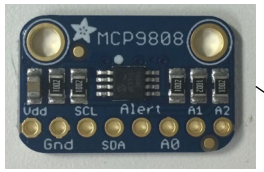


Internet

Internet

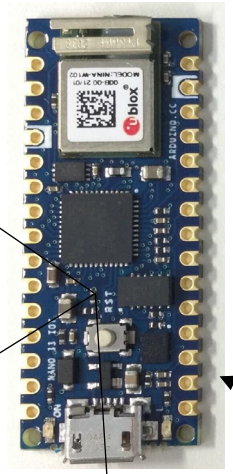
Inside your laboratory

Outside your laboratory



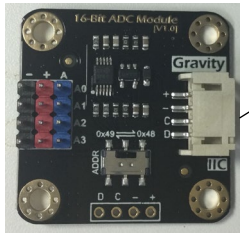
Temperature measurement

I2C



Wi-Fi

Wi-Fi router



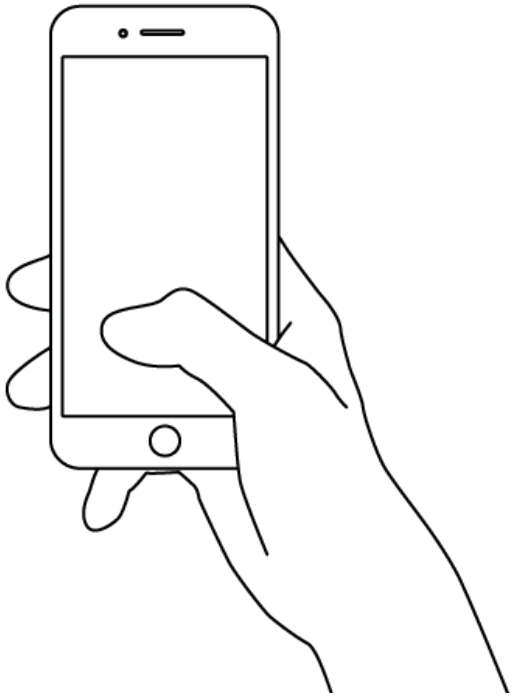
I2C

Voltage measurement
(laser power, MOT fluorescence)

I2C

External trigger for upload
data to the cloud

Displaying experimental
conditions on an OLED



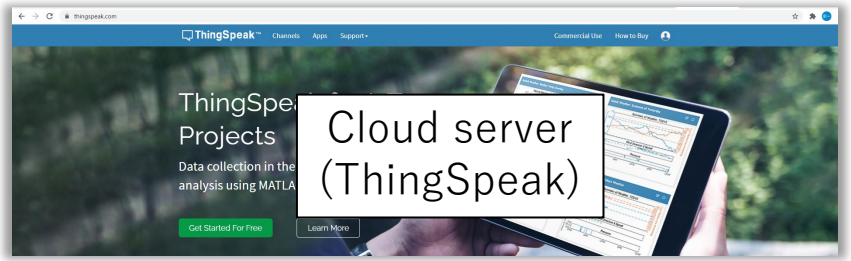
Contents

1. Arduino NANO 33 IOT
2. LED blink, Hello world
3. I2C communication
    - OLED display
    - Temperature sensor
    - ADC
4. Wi-Fi connection
5. IoT cloud server

Level : undergraduate students

This slide will be found in our laboratory HP

Today's goal

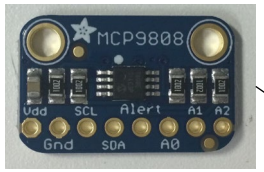


Internet

Internet

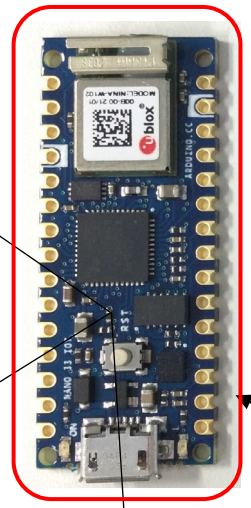
Outside your laboratory

Inside your laboratory



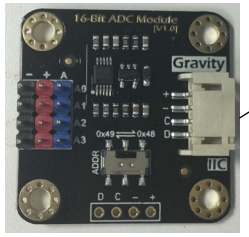
Temperature measurement

I2C



Wi-Fi

Wi-Fi router



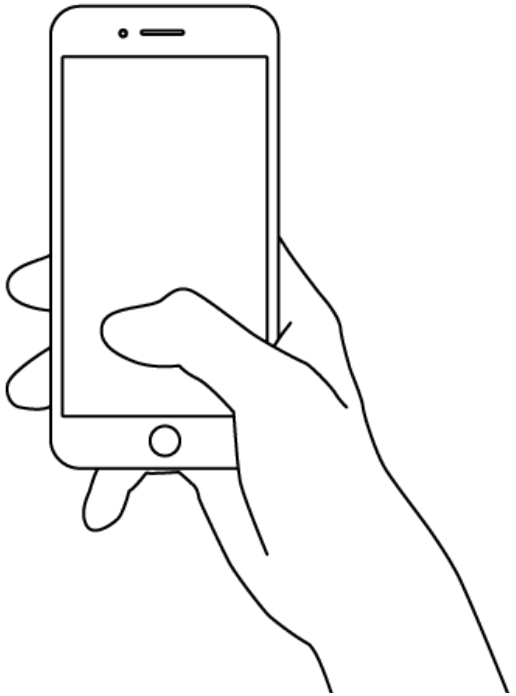
I2C

Voltage measurement
(laser power, MOT fluorescence)

External trigger for upload
data to the cloud

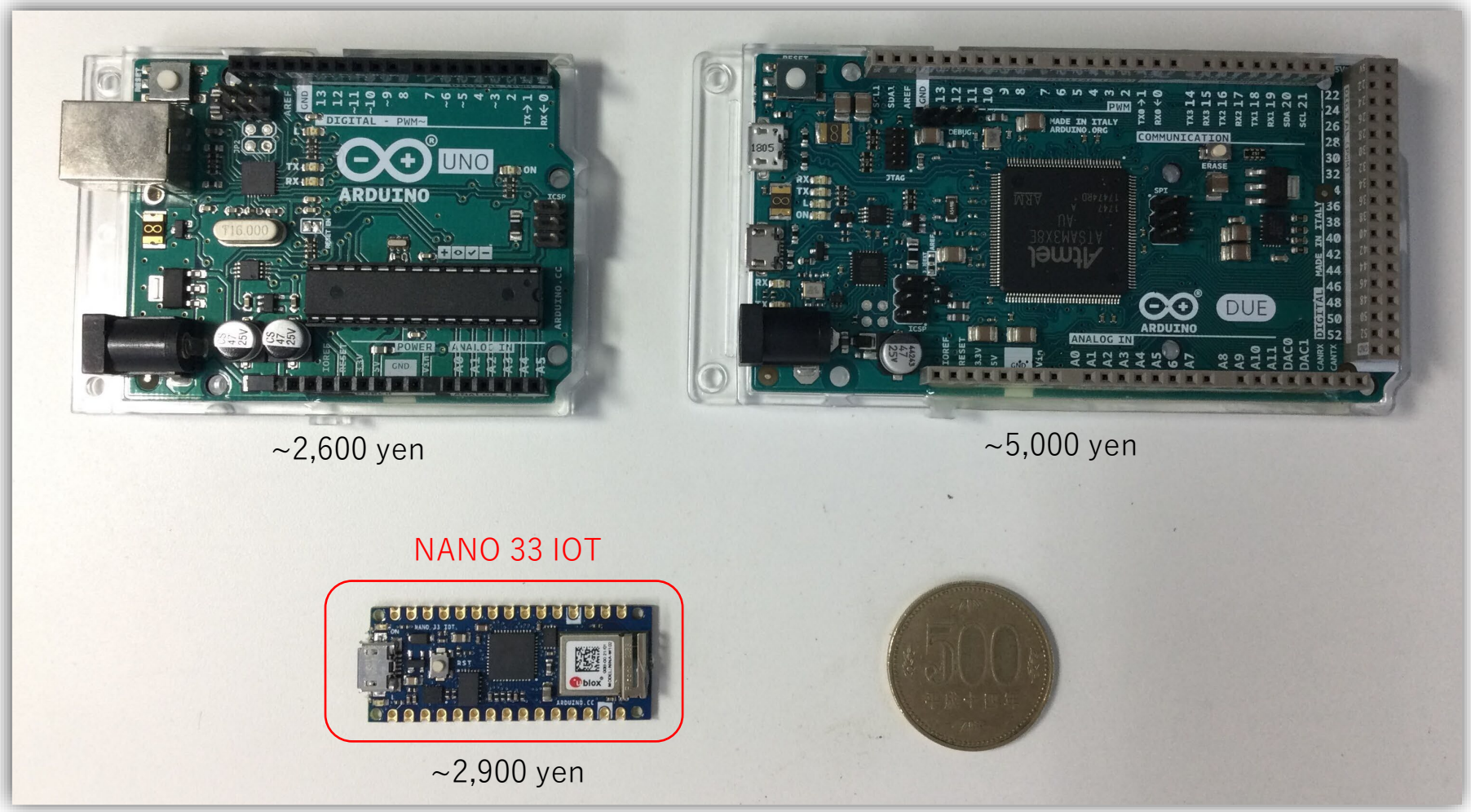
I2C

Displaying experimental
conditions on an OLED



Arduino

- Programable small computer with C-language by a free development tool
- Digital in/out, analog in, various communication
- Standalone operating system
- Low cost

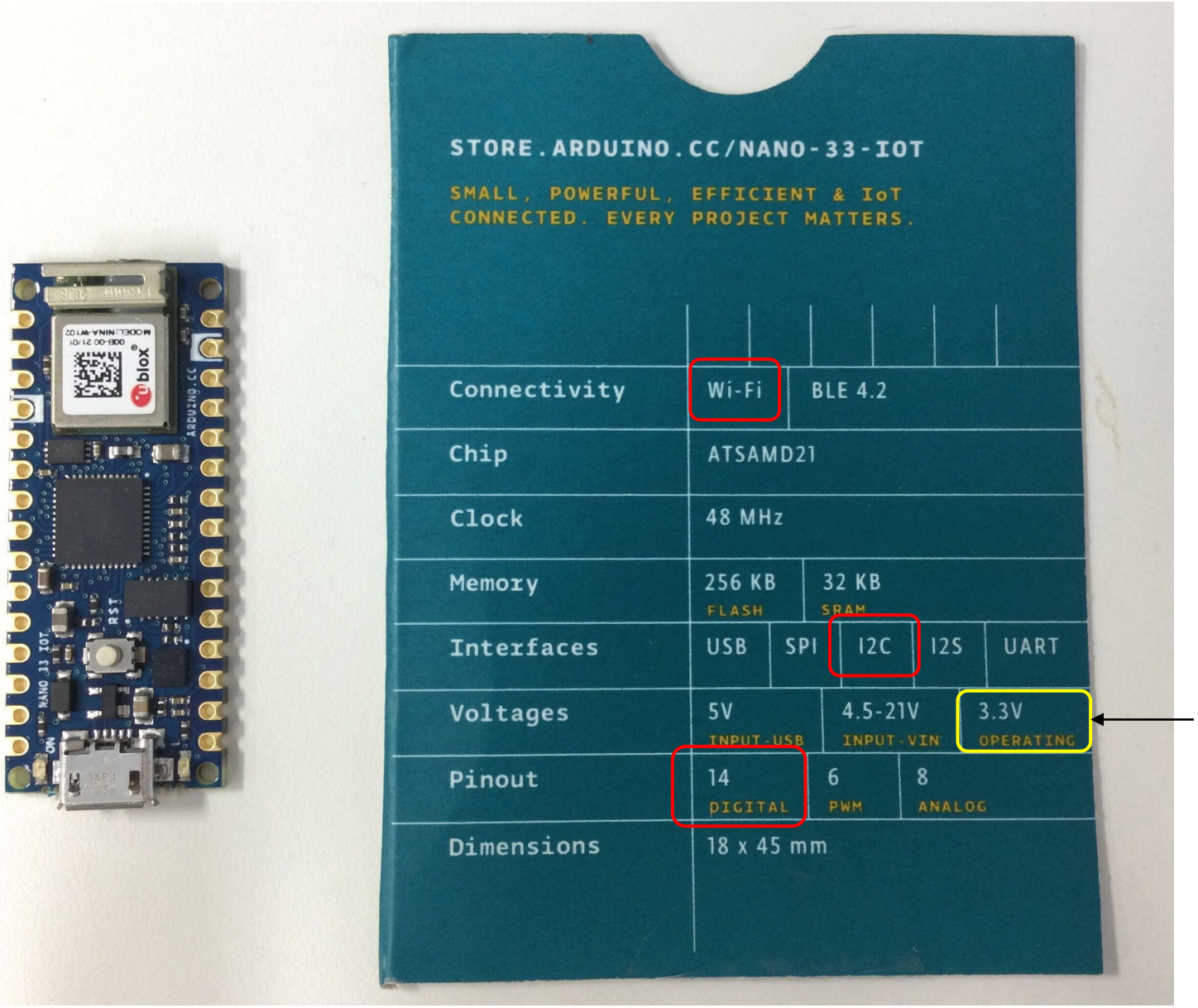


~2,600 yen

~5,000 yen

~2,900 yen

Arduino NANO 33 IOT

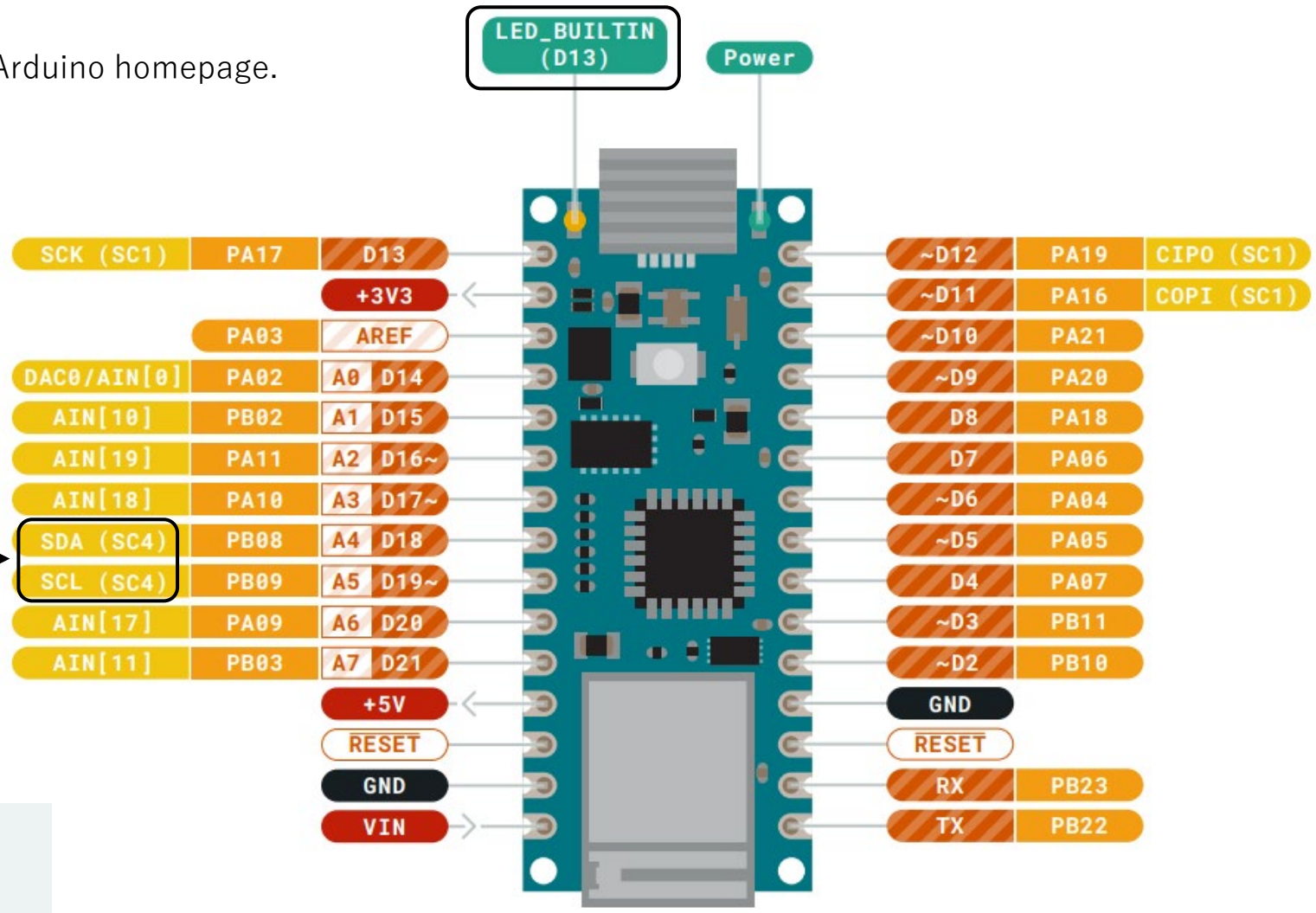


Arduino NANO 33 IOT

The full pinout diagram is available from the Arduino homepage.
https://store.arduino.cc/usa/nano-33-iot

For LED blink (Lチカ)

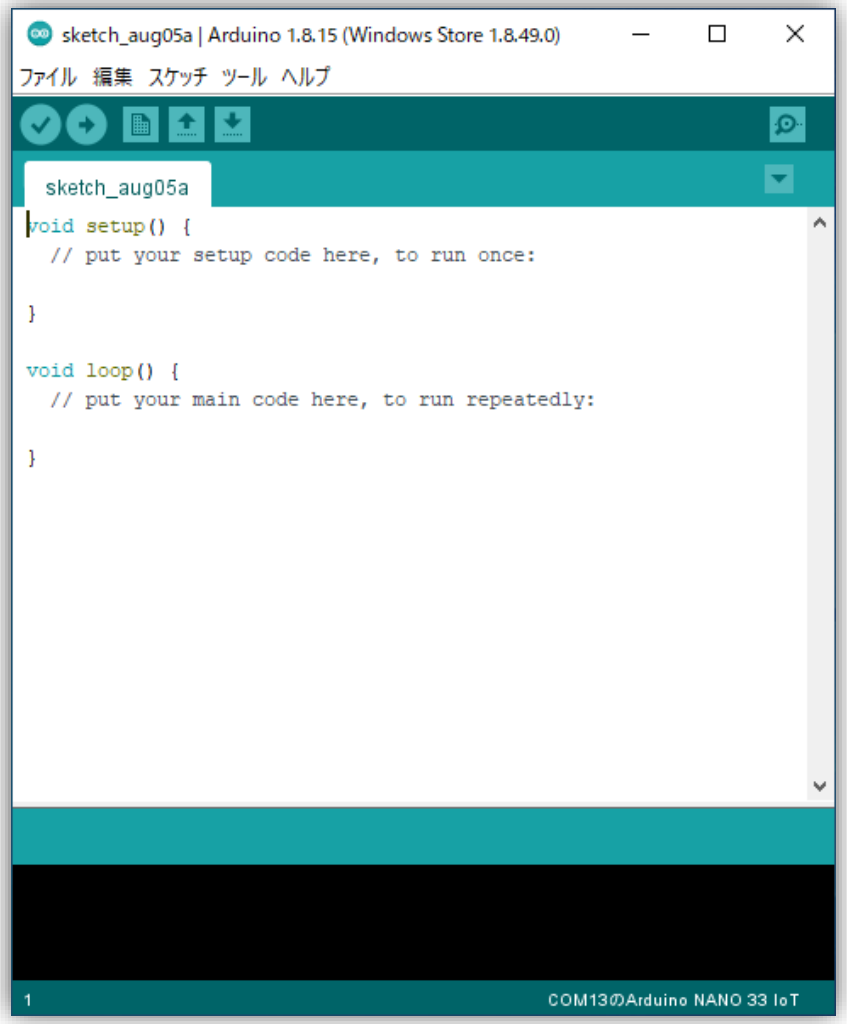
For I2C communication



Ground
Power
LED
Internal Pin
SWD Pin
Digital Pin
Analog Pin
Other Pin
Microcontroller's Port
Default

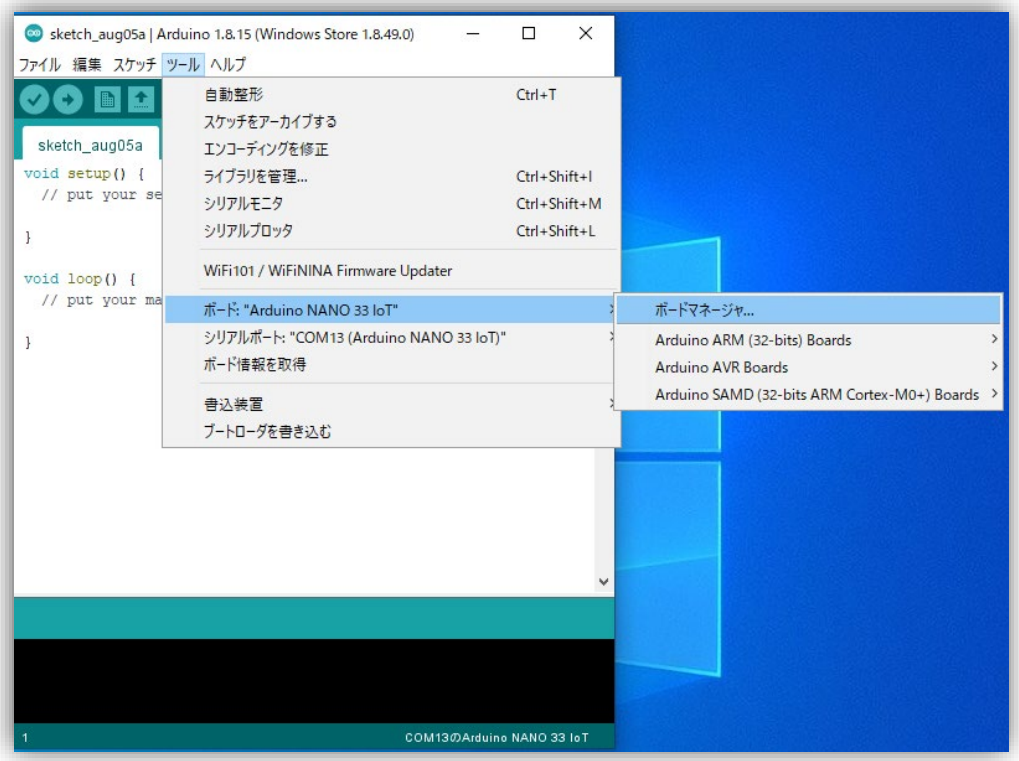
How to program it

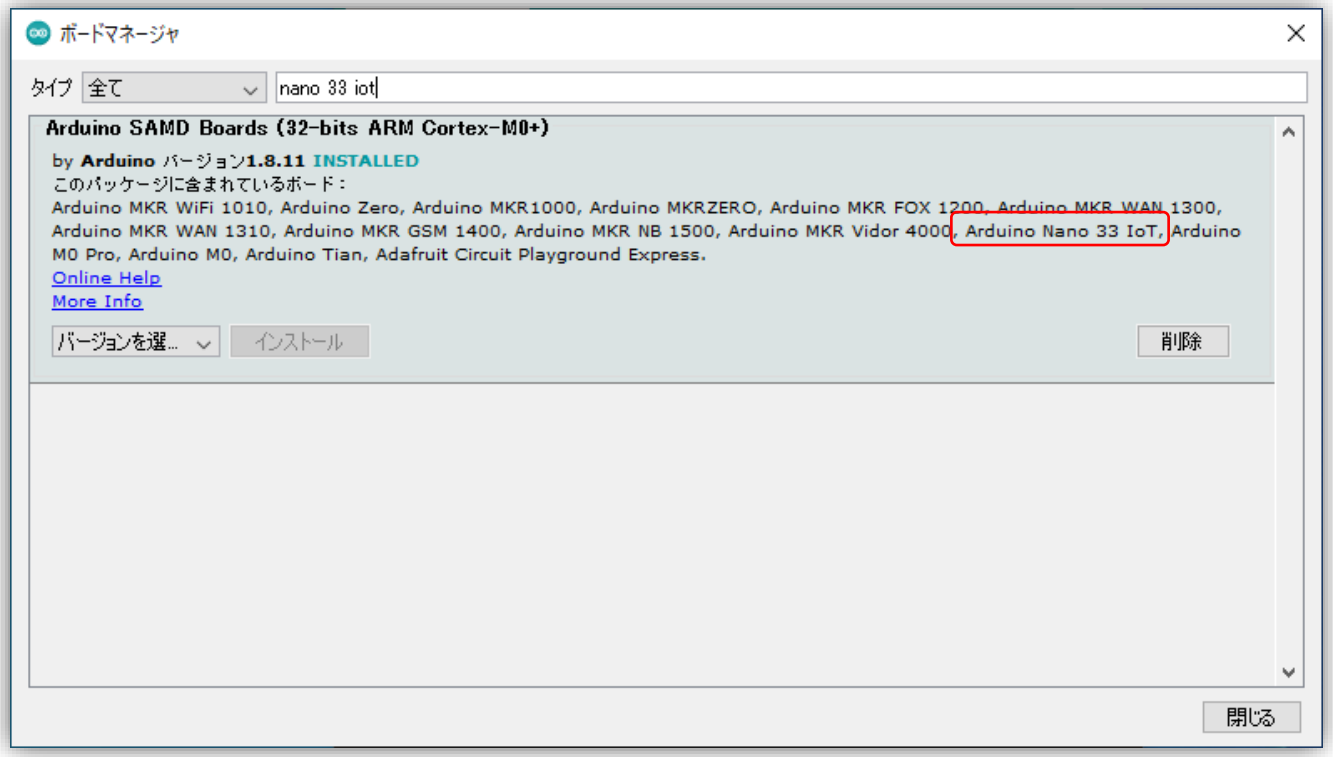
- Install Arduino IDE (free) to your PC and launch it (https://www.arduino.cc/en/software)



How to program it

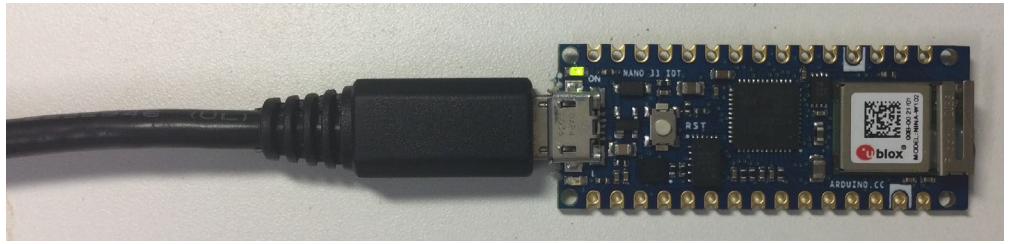
- Open board manager and type ‘nano 33 iot’, and install a package for the CPU



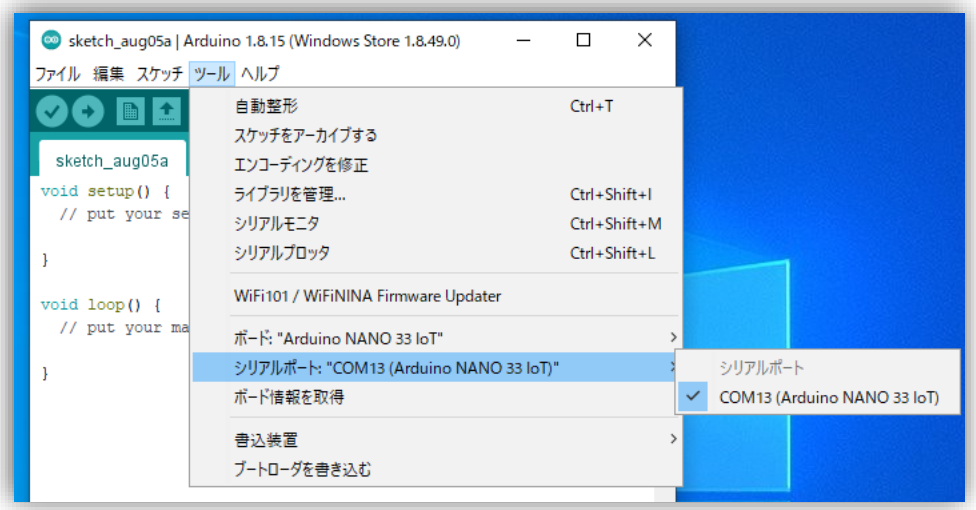


How to program it

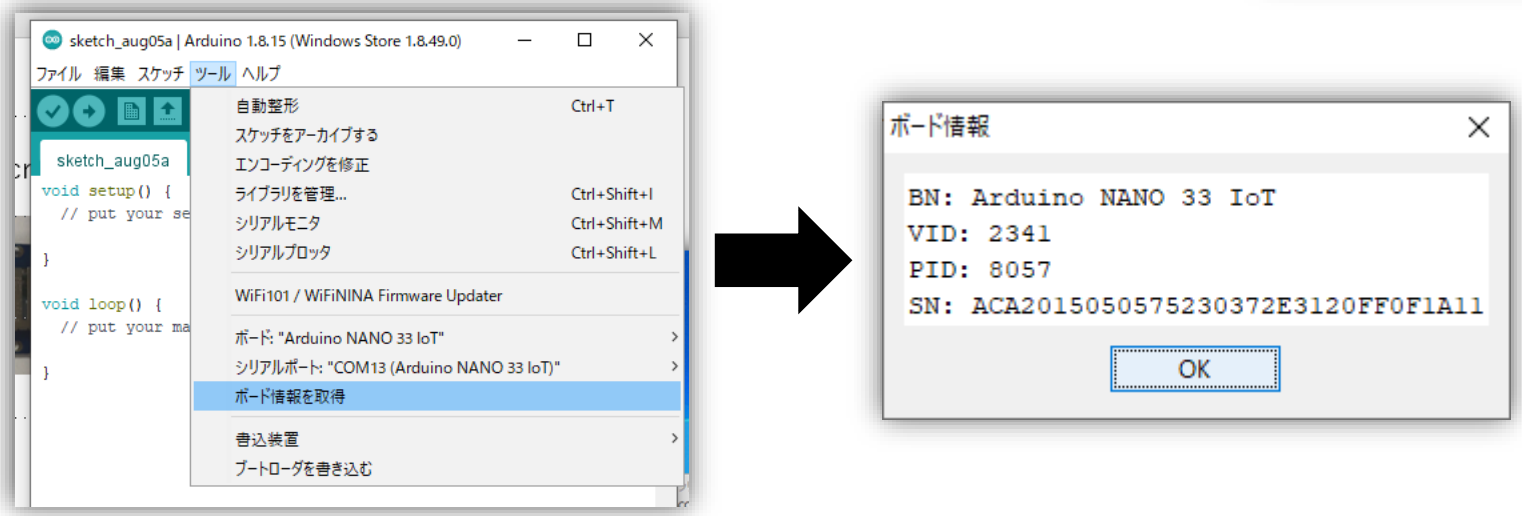
- Connect your Arduino to the PC by a USB cable(micro-B type)



- Select a serial port connecting to the Arduino

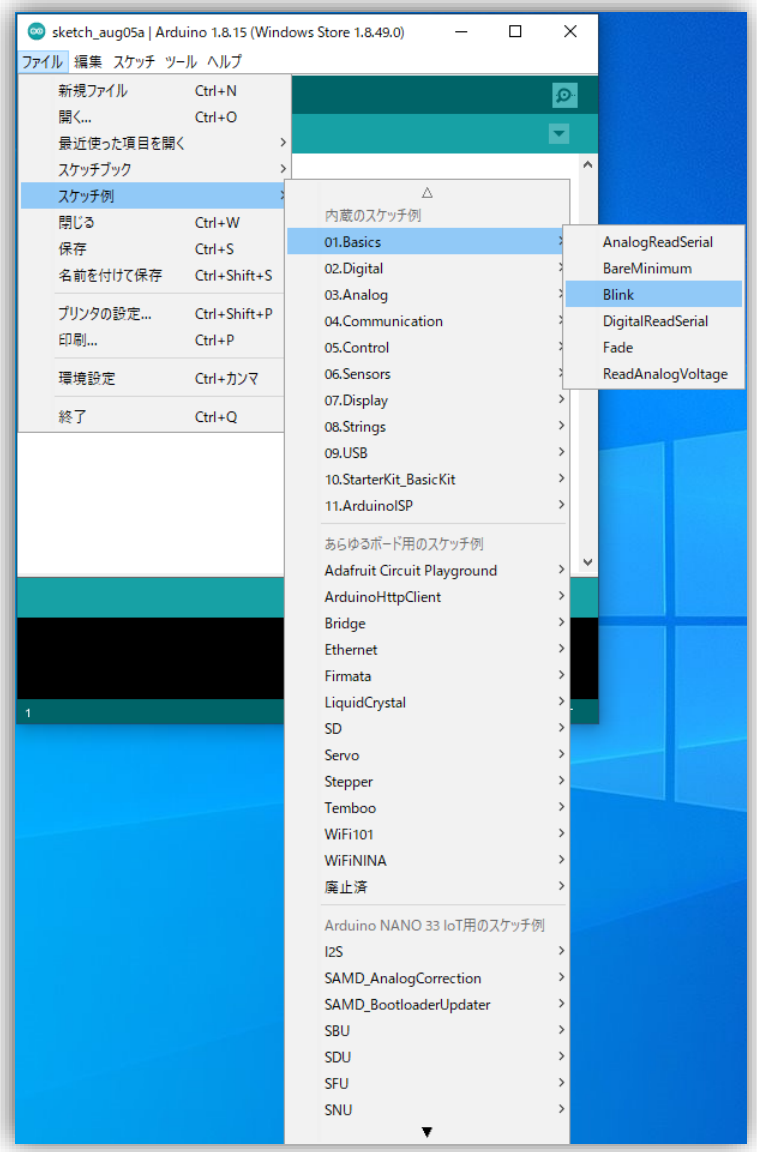


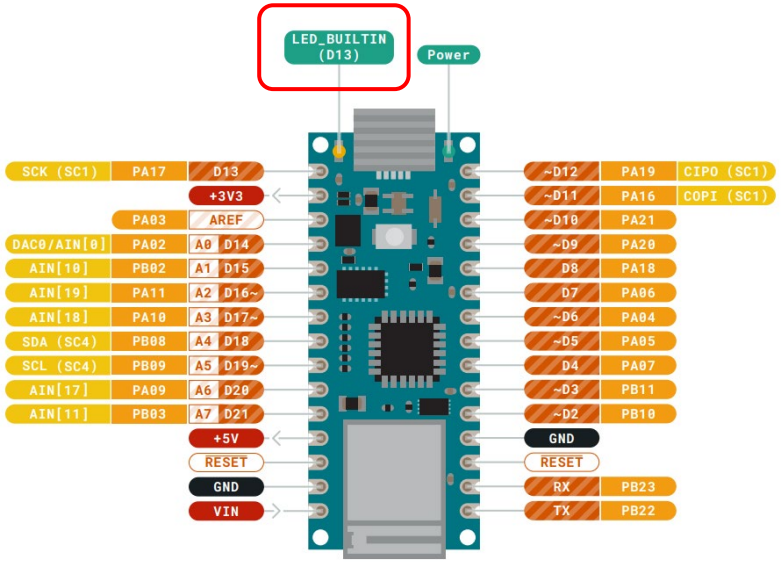
- Confirm your Arduino

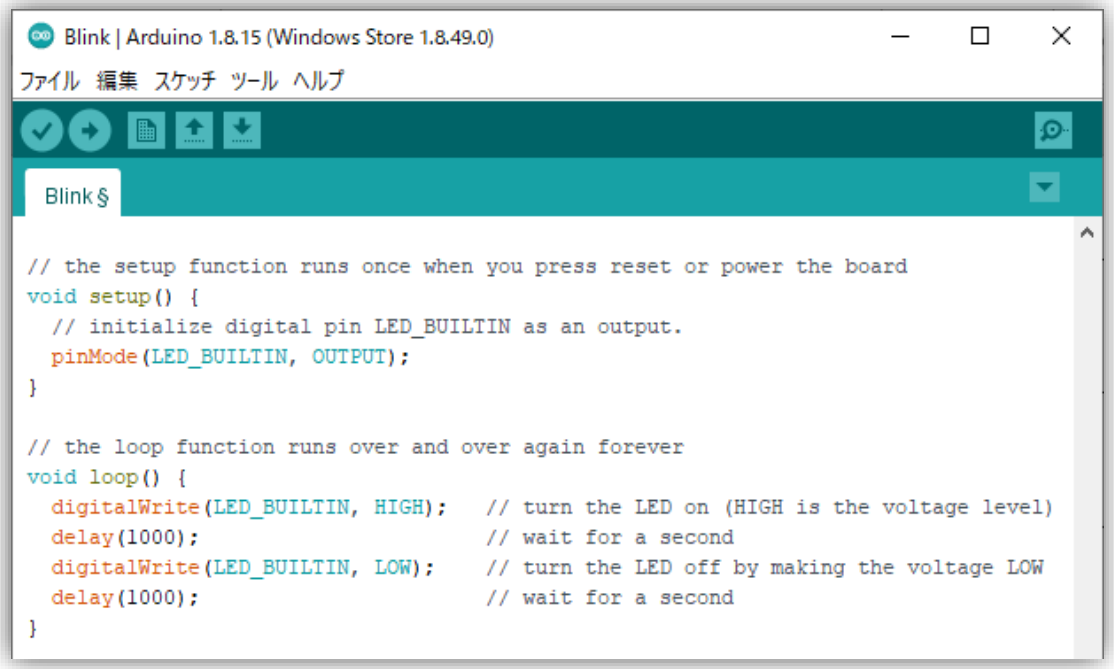


LED blink

- Open a sample code

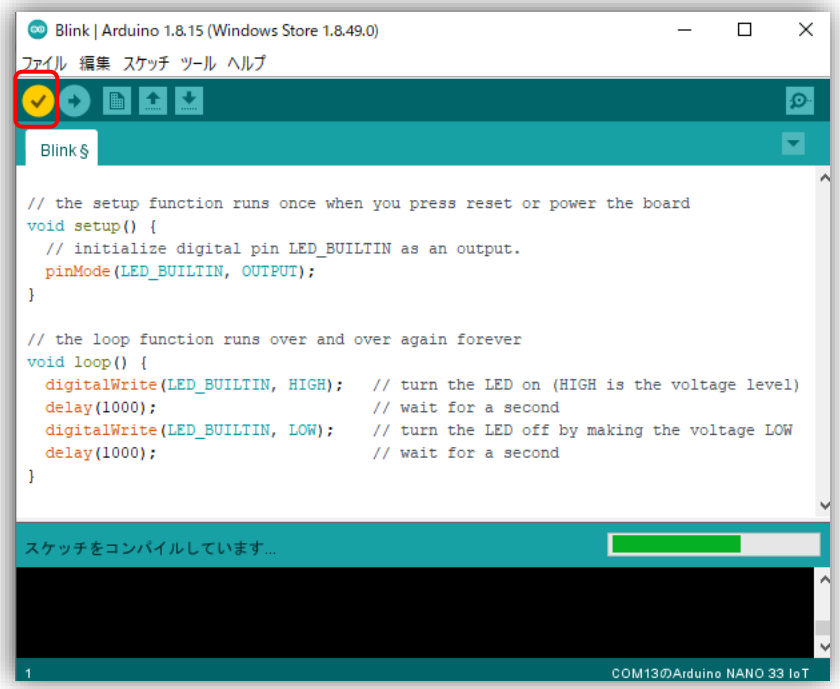




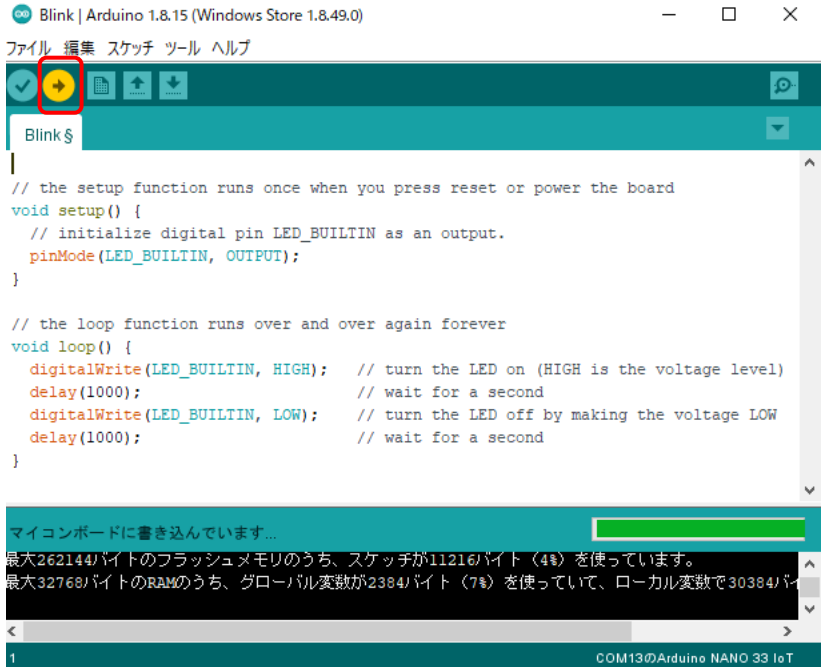


LED blink

- Compile



- Write it to your Arduino



LED blink

Change the output channel and the period

Blink | Arduino 1.8.15 (Windows Store 1.8.49.0)

ファイル 編集 スケッチ ツール ヘルプ

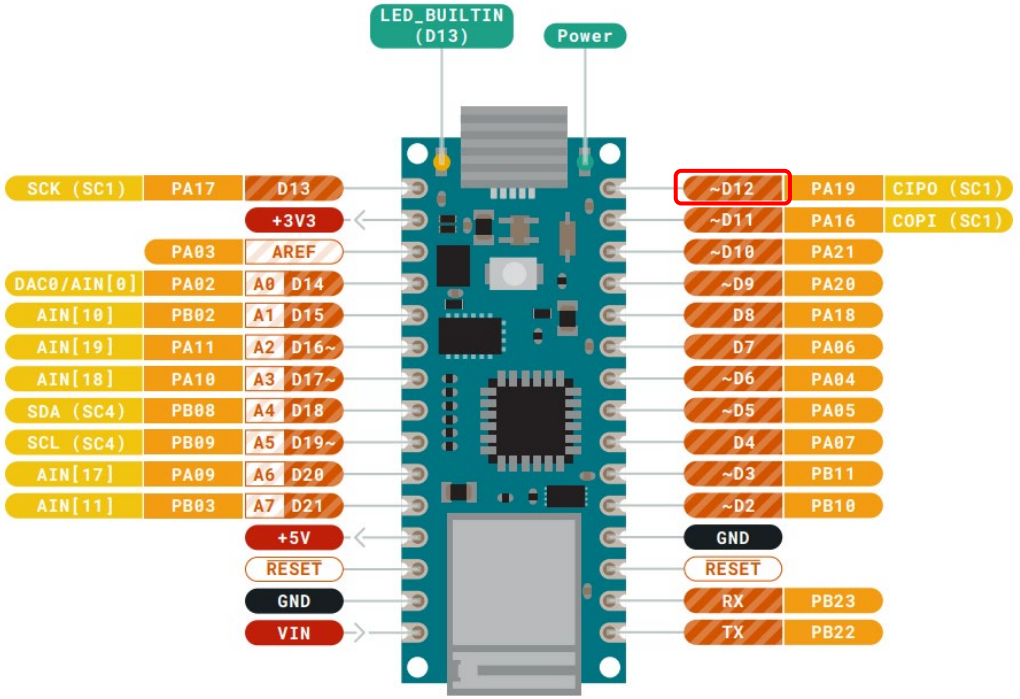
Blink §

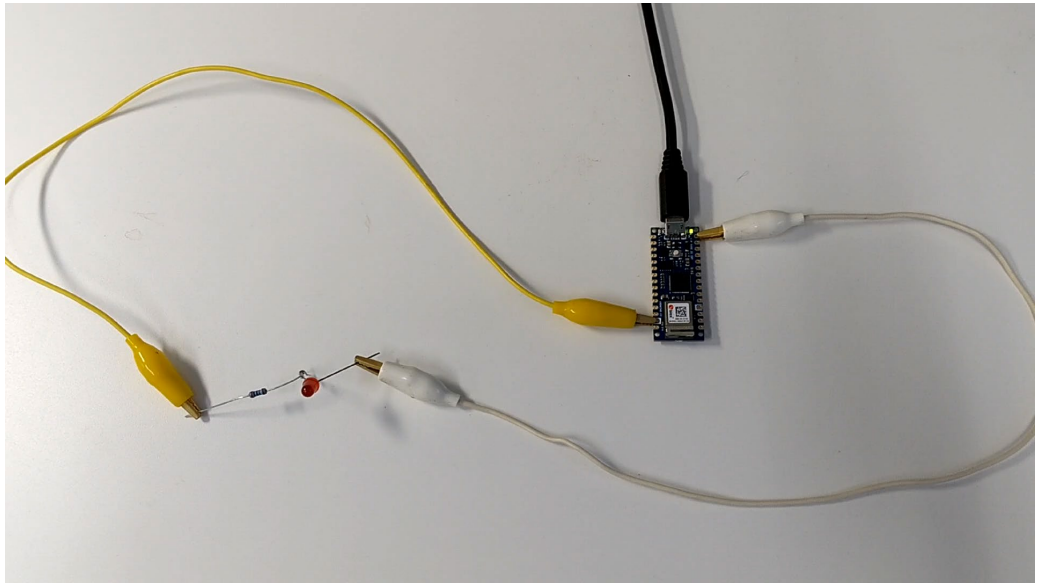
```
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(12, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(12, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(500);                       // wait for a 0.5 second
  digitalWrite(12, LOW);    // turn the LED off by making the voltage LOW
  delay(500);                       // wait for a 0.5 second
}
```

ボードへの書き込みが完了しました。

```
done in 0.011 seconds
CPU reset.
```

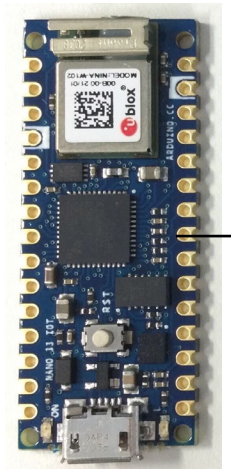
11 COM13のArduino NANO 33 IoT





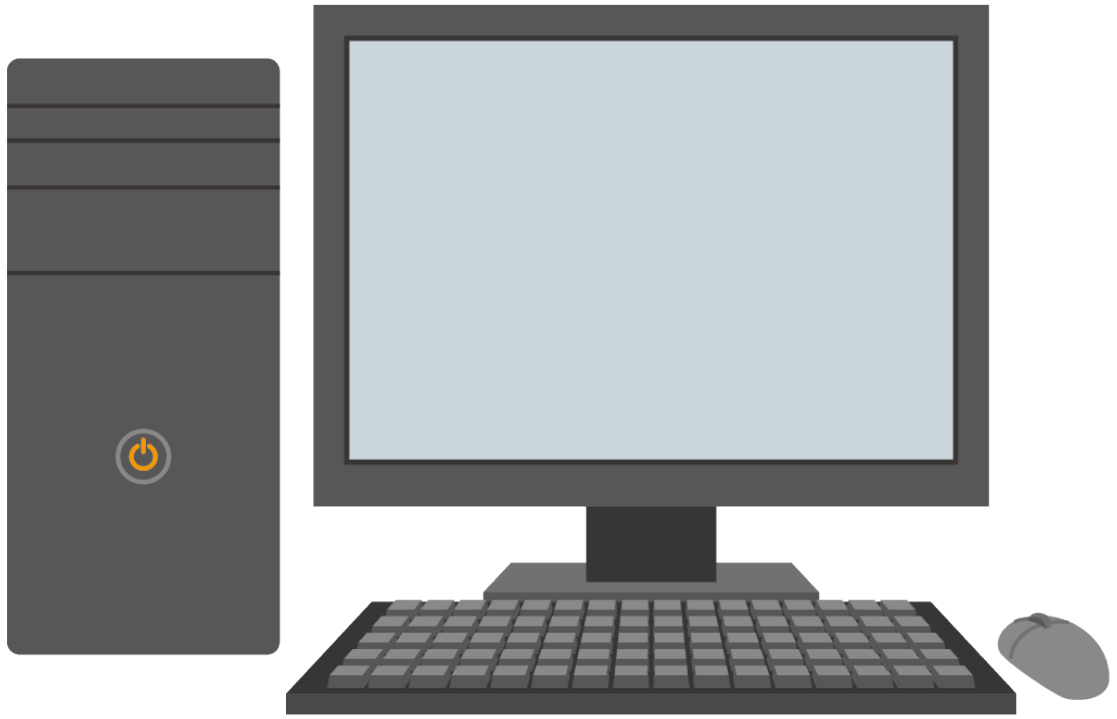
Hello world

Serial communication



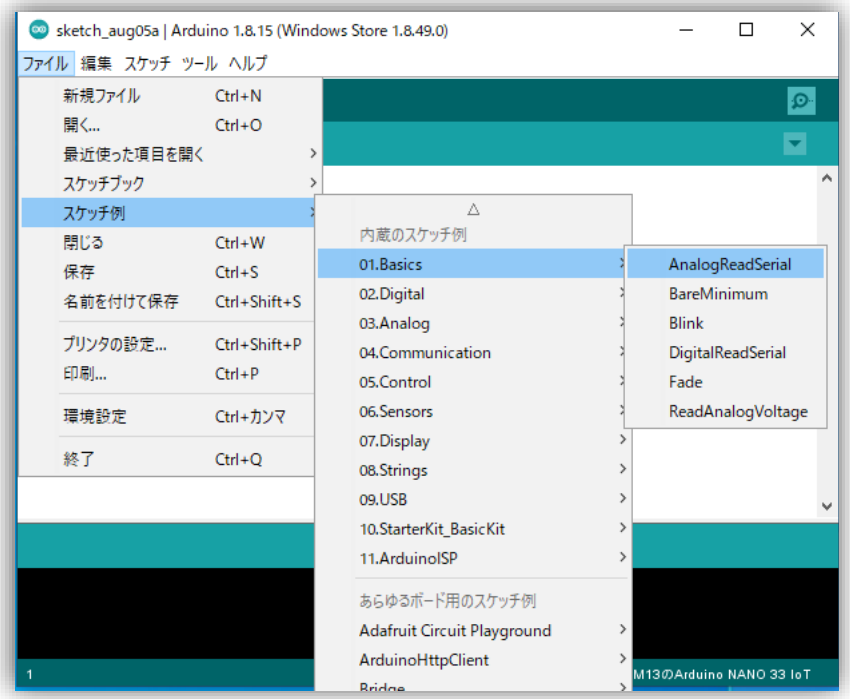
‘Hello world’

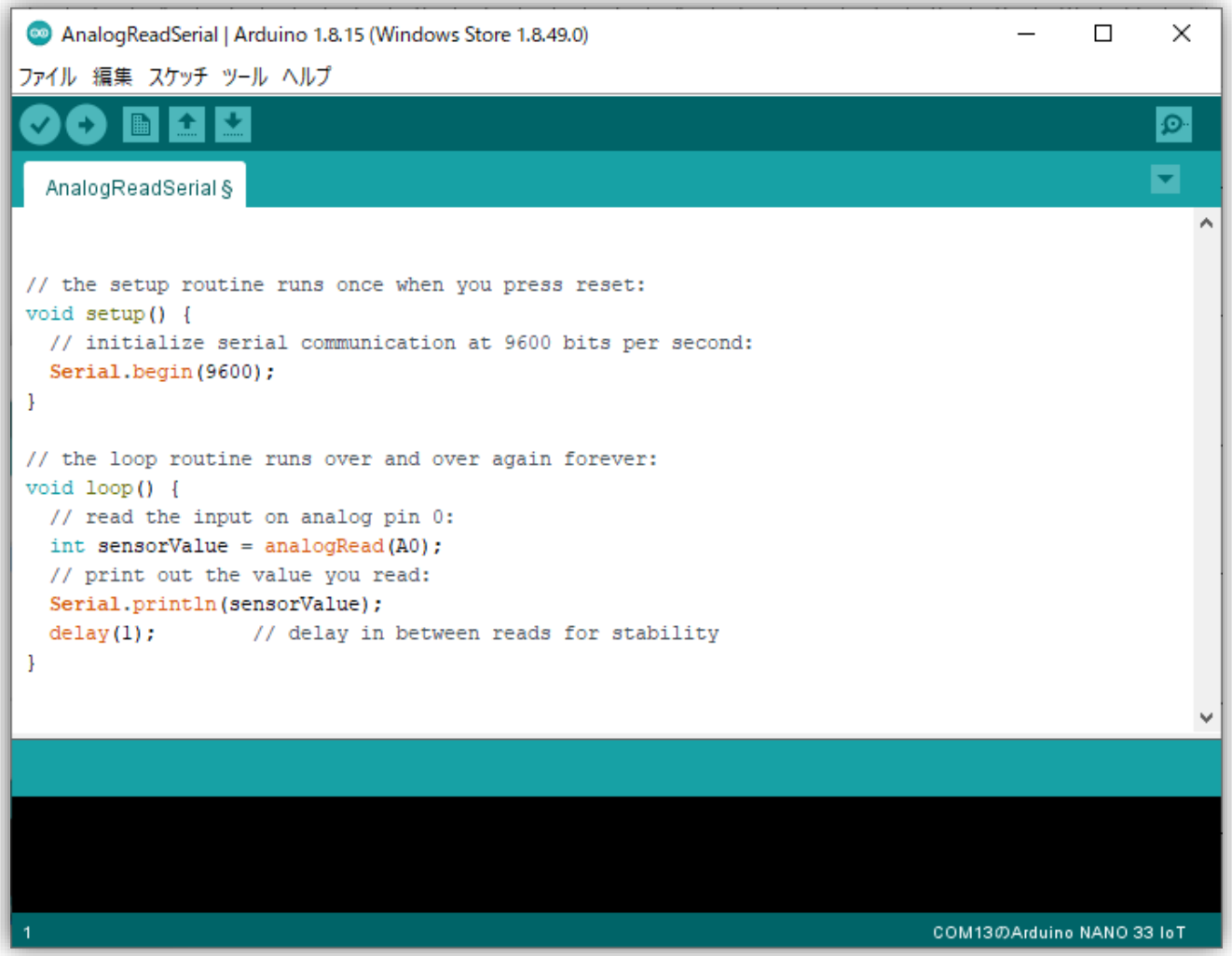
(USB cable)



Hello world

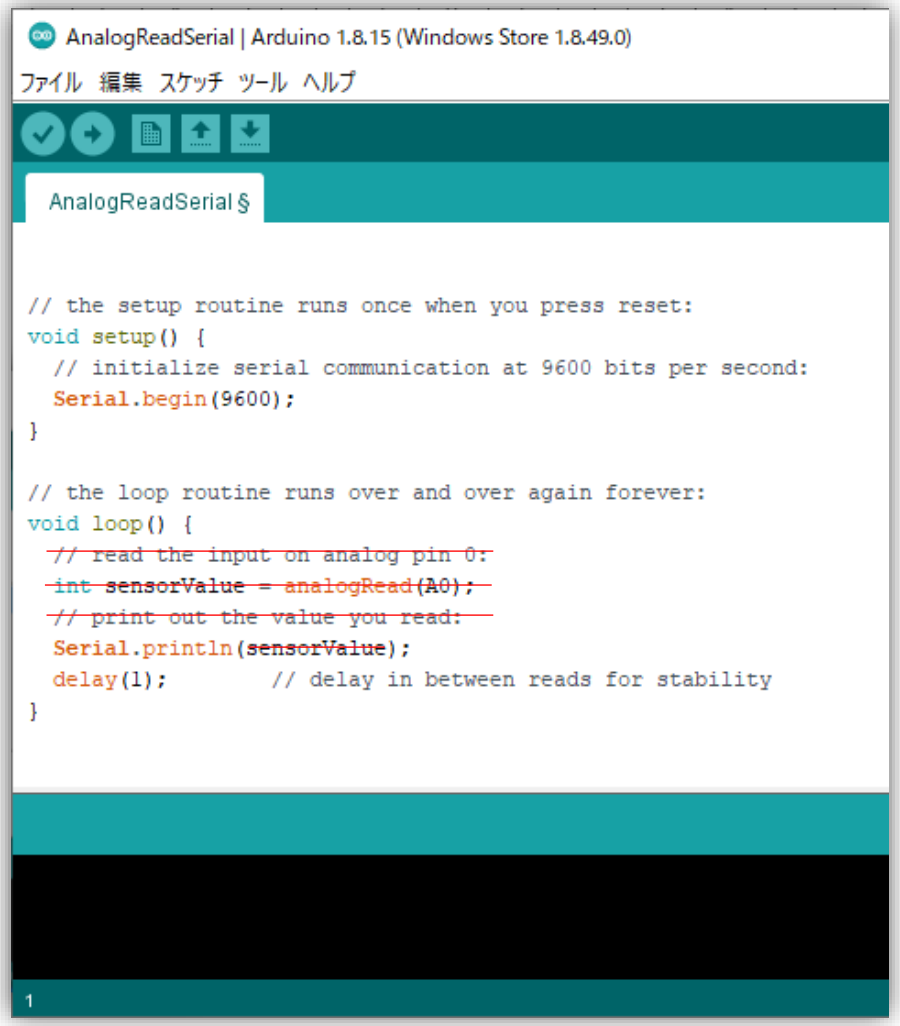
- Open a sample code



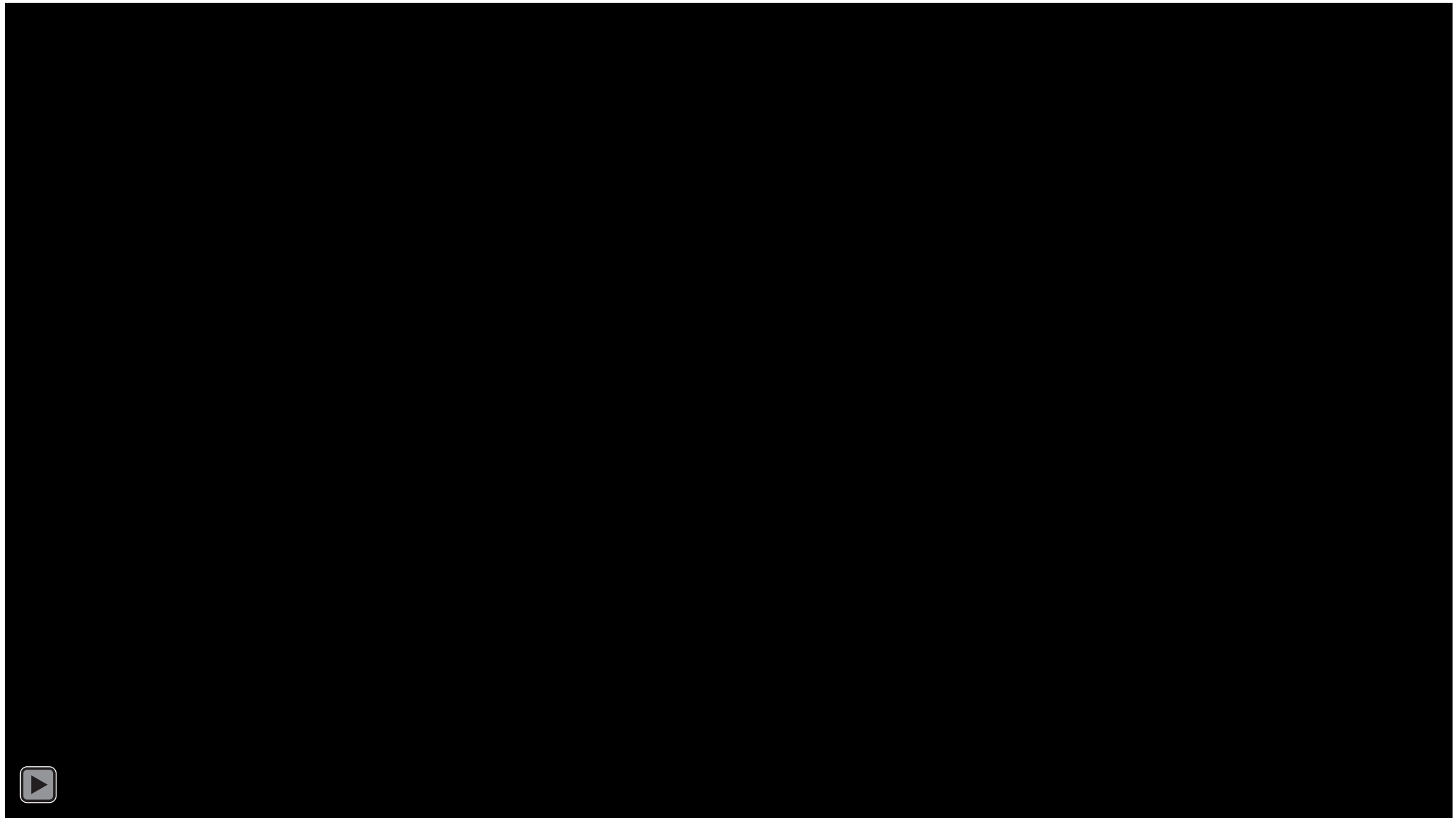


Hello world

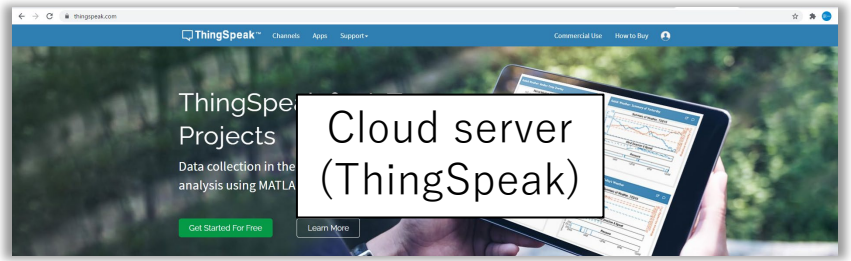
- Edit the code



Hello world



Today's goal



Internet

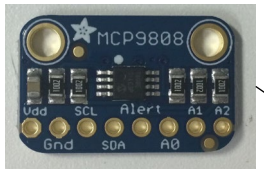
Internet

Inside your laboratory

Outside your laboratory

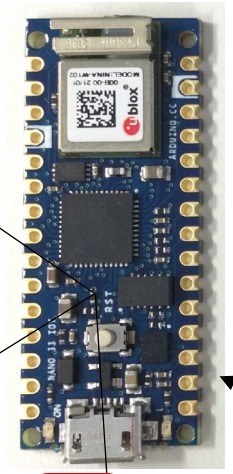
Wi-Fi

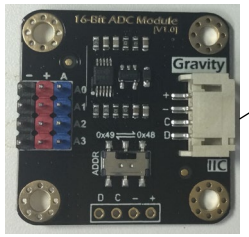
Wi-Fi router



Temperature measurement

I2C





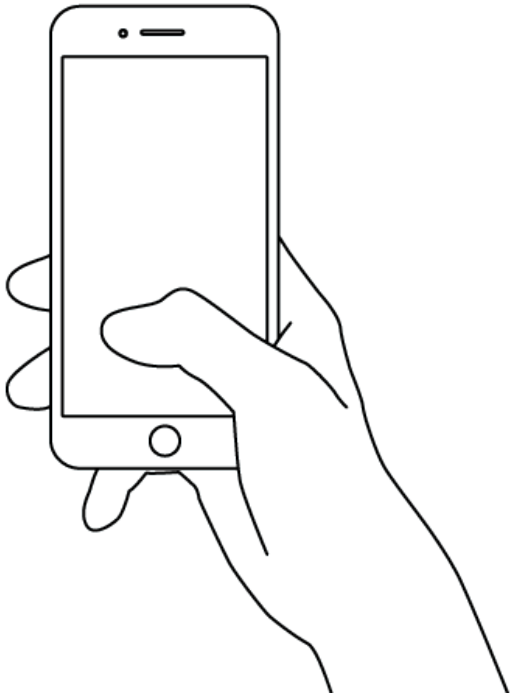
I2C

Voltage measurement
(laser power, MOT fluorescence)

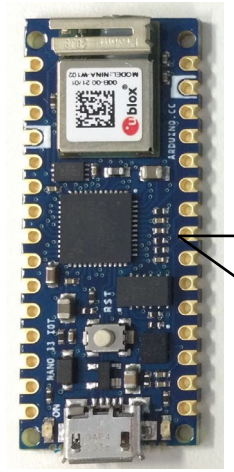
I2C

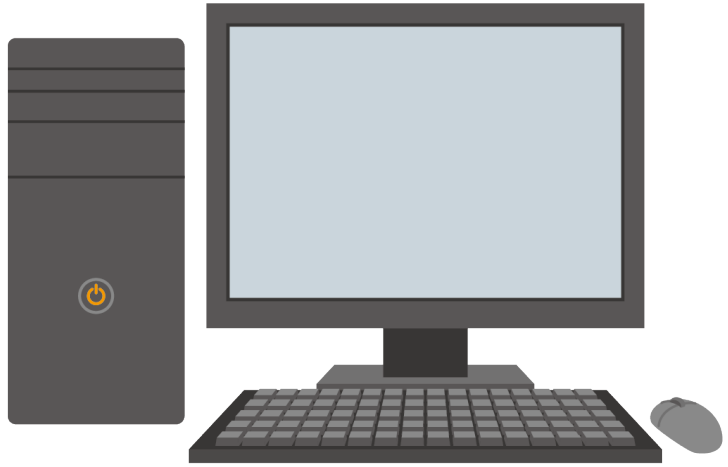
Displaying experimental
conditions on an OLED

External trigger for upload
data to the cloud

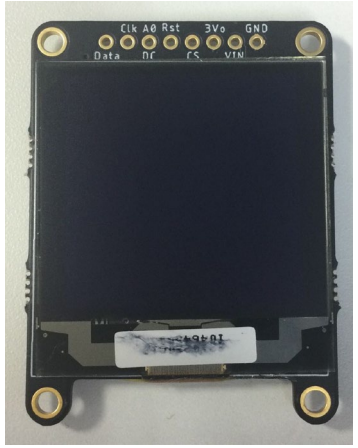


Change the display for standalone operating



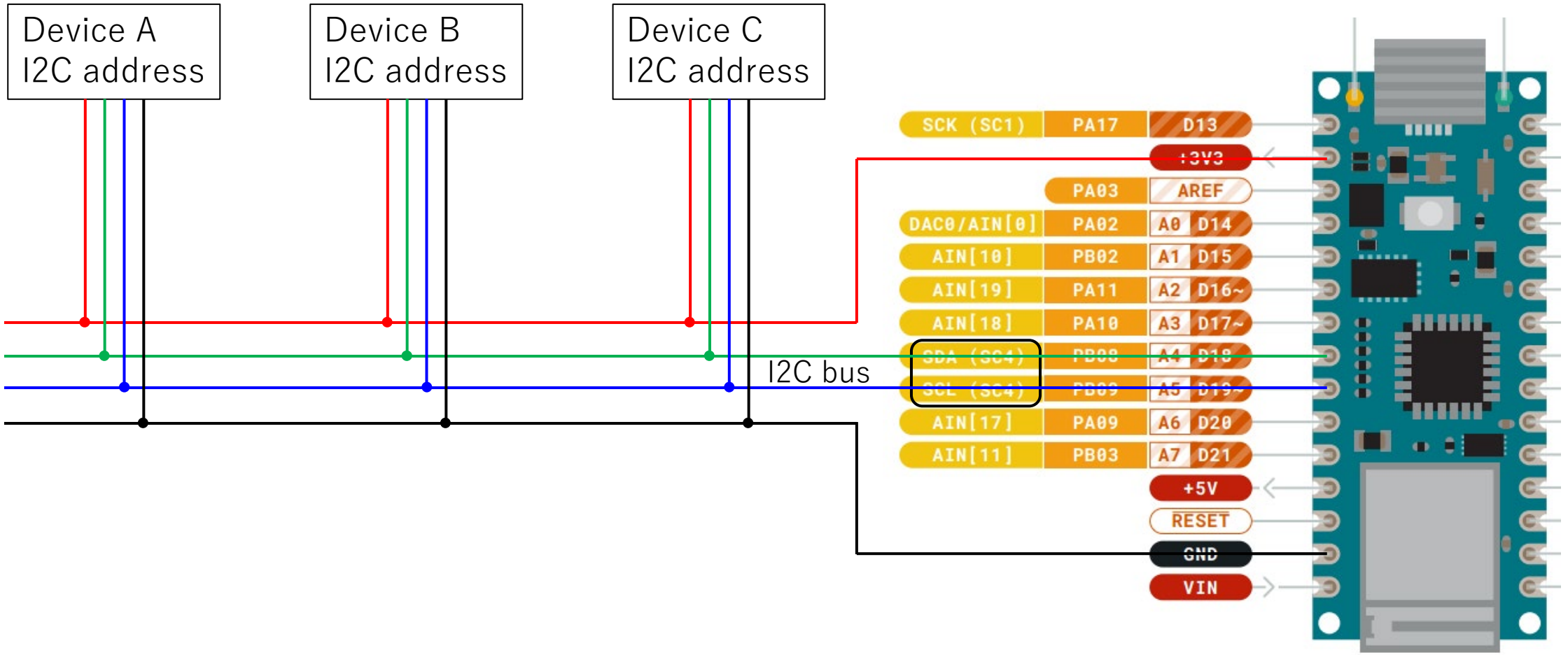


‘Hello world’
(I2C communication)



OLED
(Organic Light Emitting Diode)

I2C communication



Various digital devices with I2C communication

Grayscale 1.5" 128x128 OLED Display

‘ADAFRUIT SSD1327’

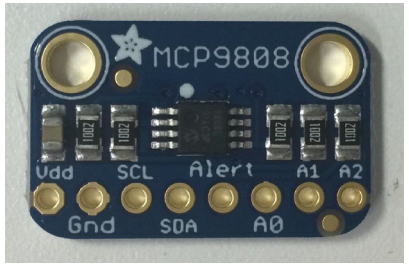
~3,000 yen

Accuracy Temperature Sensor

‘Adafruit MCP9808’

- Up to 8 on a single I2C bus with adjustable address pins
- 0.25° C typical precision over -40°C to 125°C range
- 0.0625° C resolution

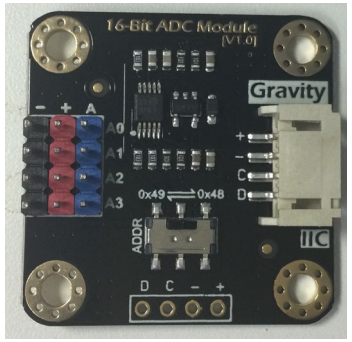
~700 yen



ADS1115 16-Bit ADC - 4 Channel with Programmable Gain Amplifier

‘DFR0553’

~1,400 yen

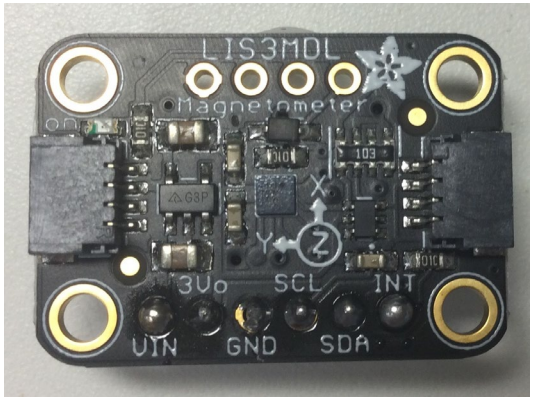


Triple-axis Magnetometer

‘Adafruit LIS3MDL’

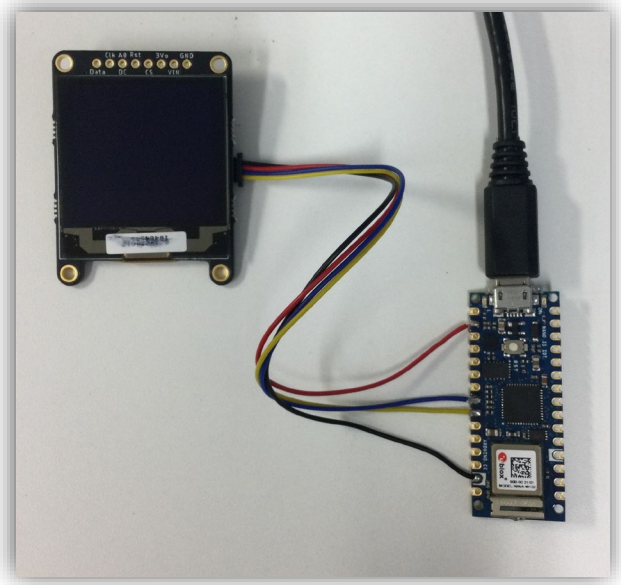
- ±4/±8/±12/±16 gauss selectable magnetic full scales
- 16-bit data output

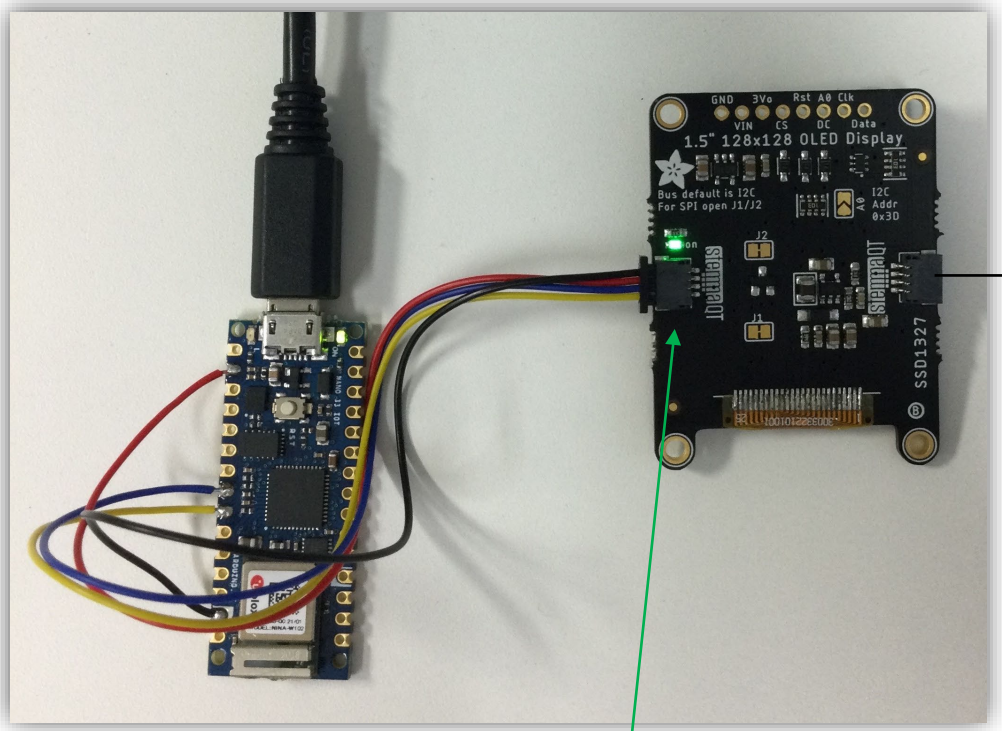
~1,000 yen



Change the display for standalone operating

- I2C connection



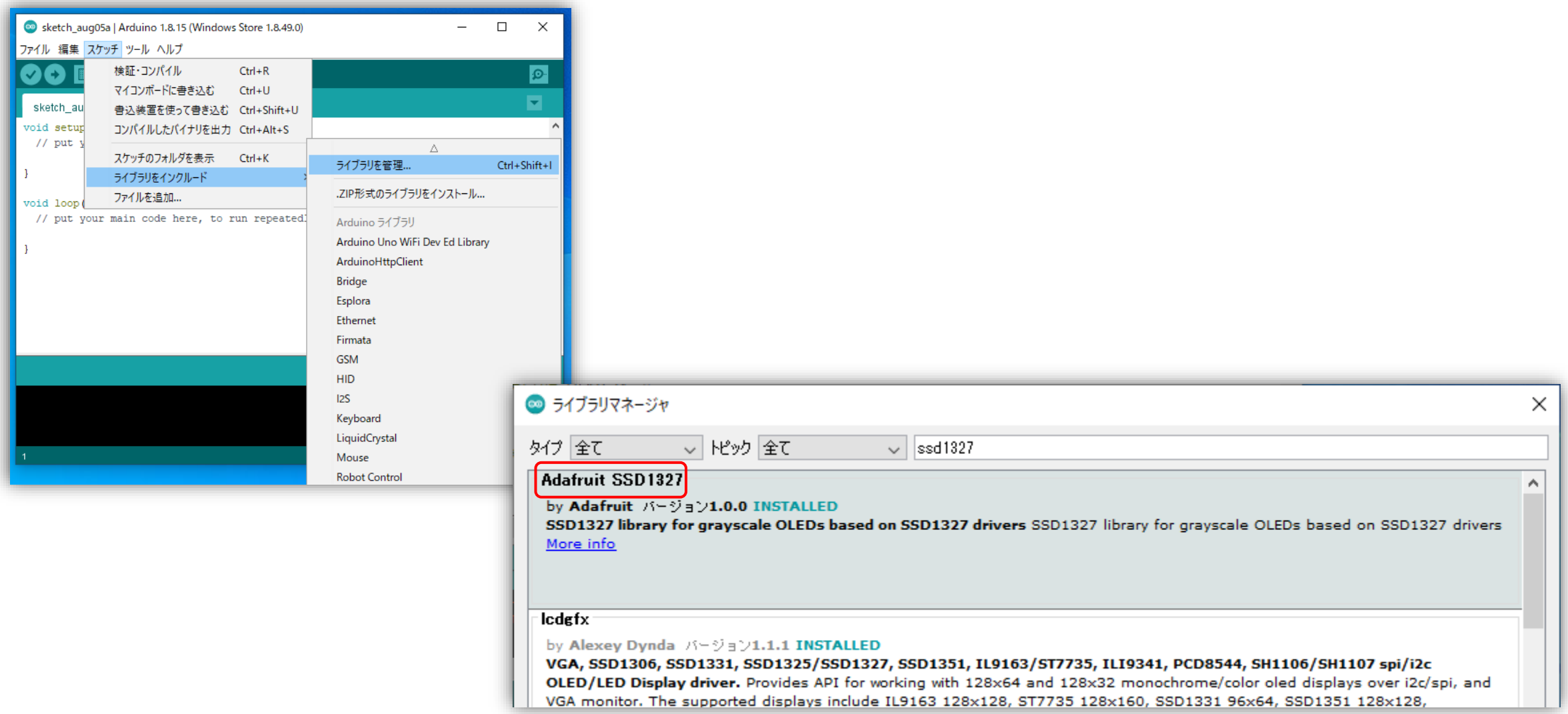


Additional I2C devices

STEMMA QT / Qwiic connector

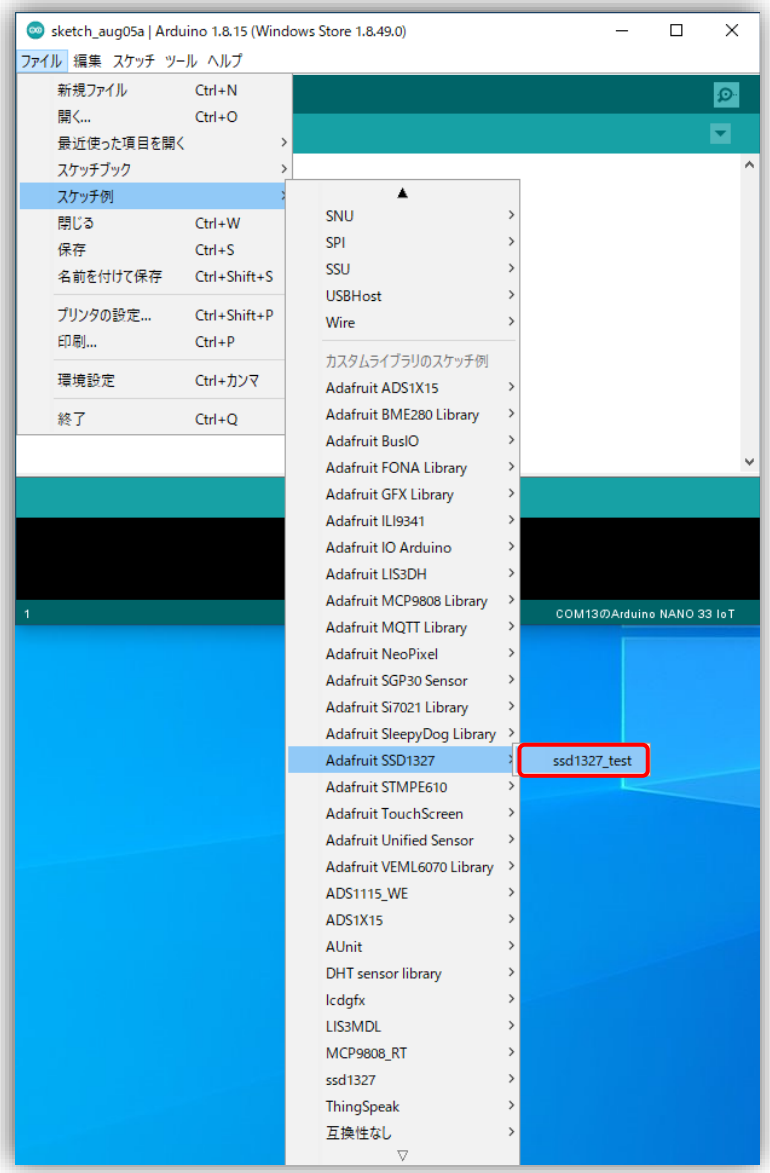
Change the display for standalone operating

- Install the library of SSD1327

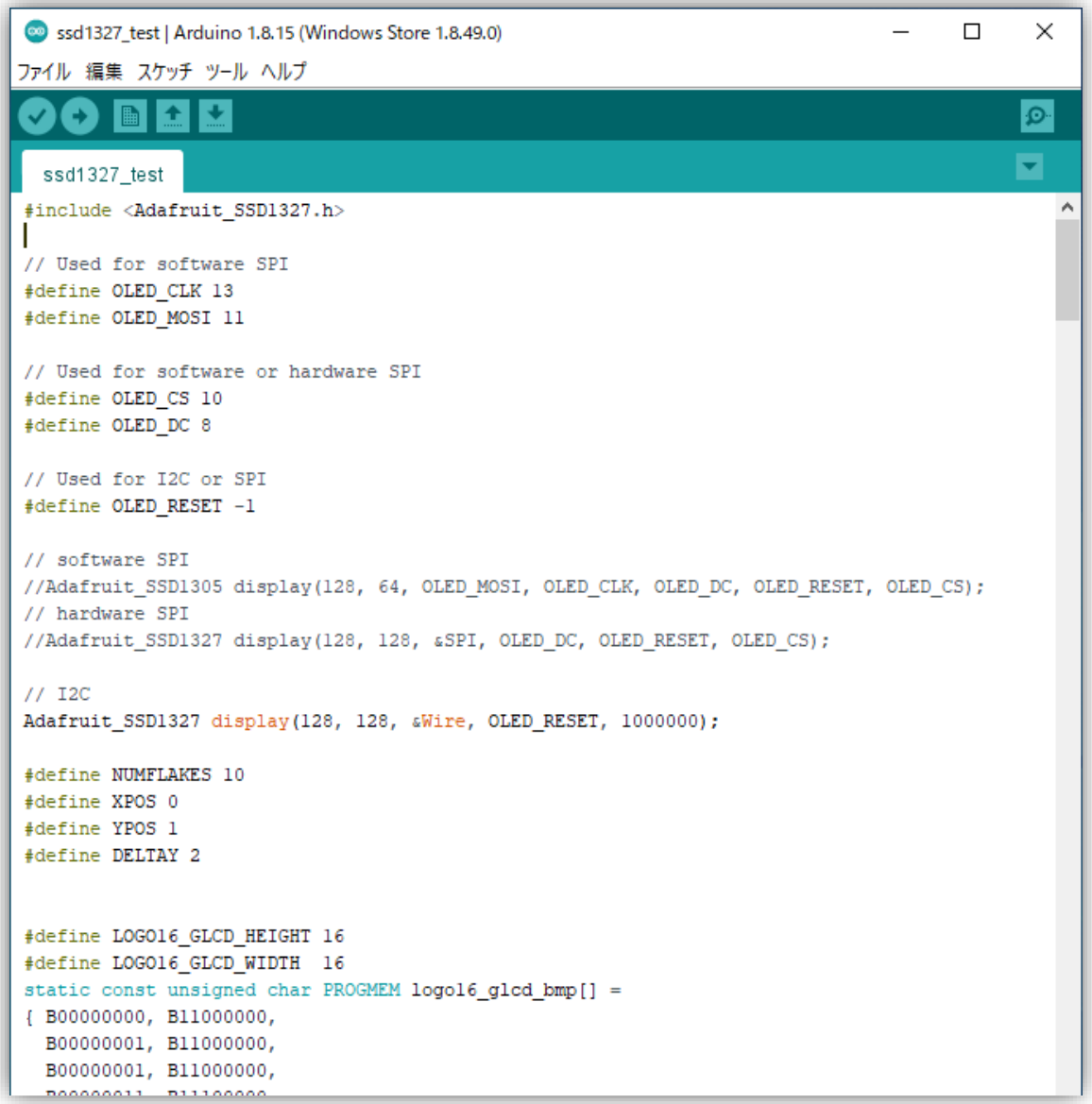


Change the display for standalone operating

- Open a sample code

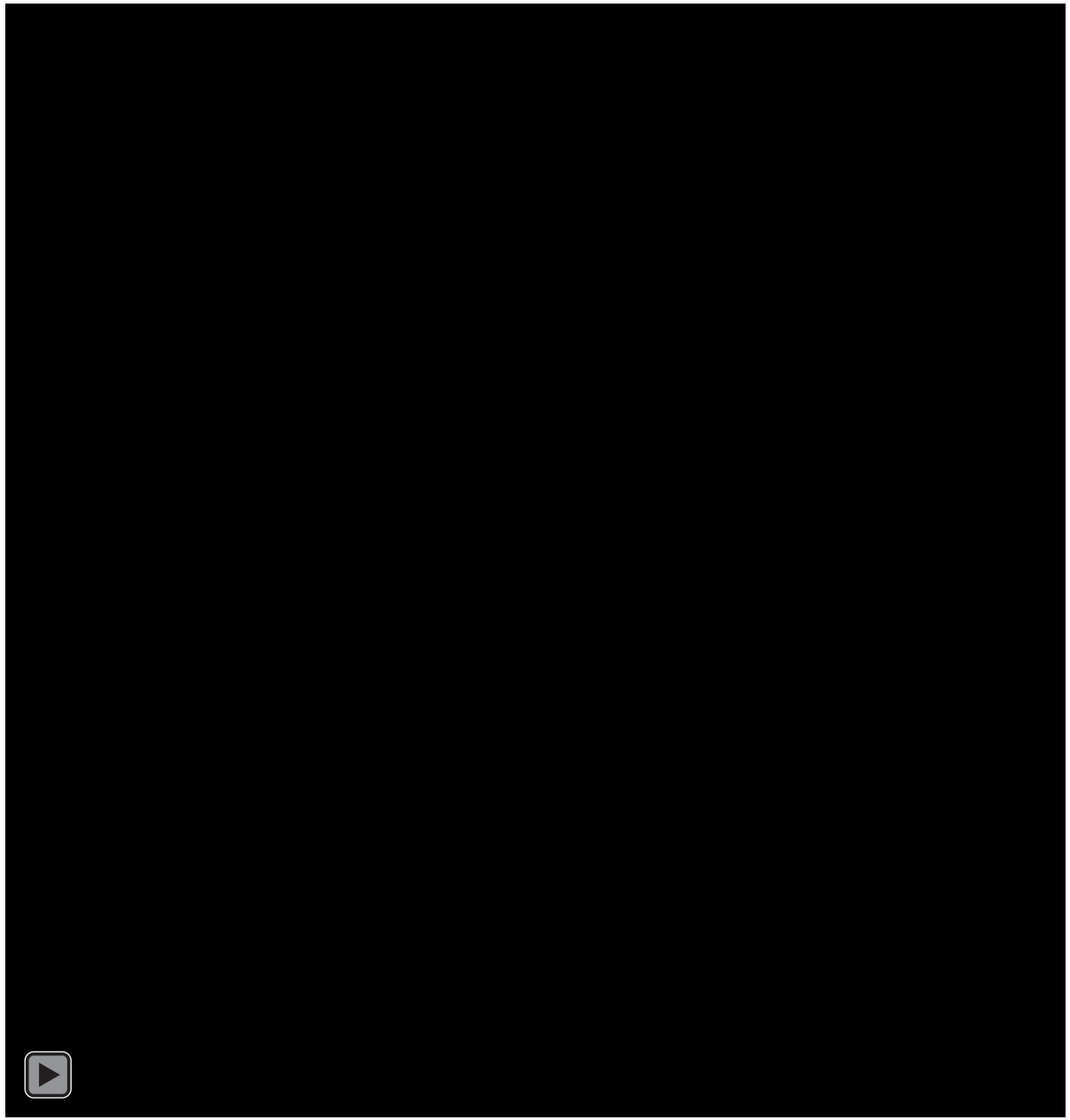


- Write it to your Arduino



Change the display for standalone operating

- Test run

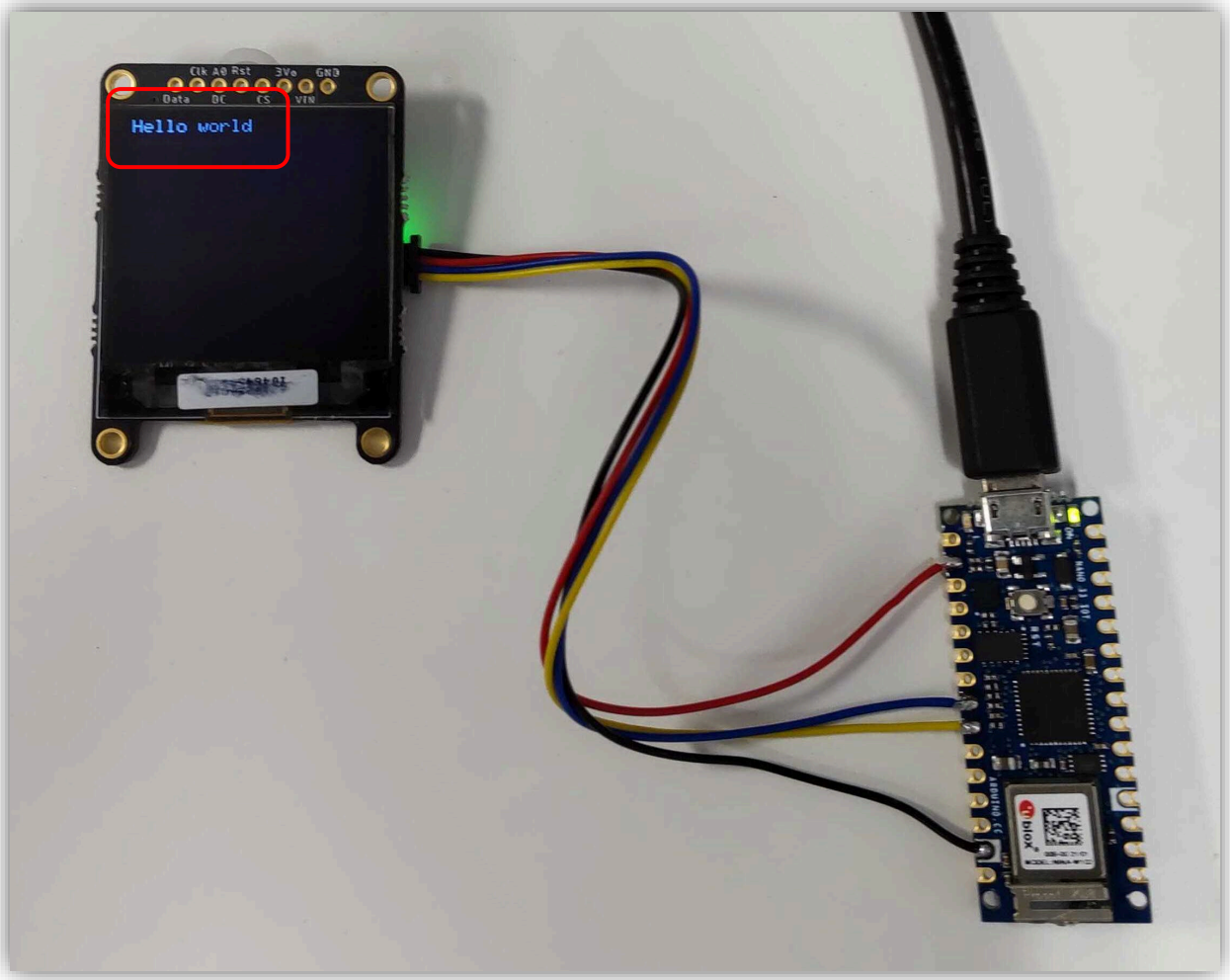


Change the display for standalone operating

- Cut and paste

Hello

```
#include <Adafruit_SSD1327.h>

// I2C
#define OLED_RESET -1
Adafruit_SSD1327 display(128, 128, &Wire, OLED_RESET, 1000000);

// the setup routine runs once when you press reset:
void setup() {
  display.begin(0x3D);
  display.clearDisplay();
  display.display();
}

// the loop routine runs over and over again forever:
void loop() {
  String s = "Hello world";

  display.clearDisplay();
  display.setTextSize(1);
  display.setCursor(0, 0);

  display.println(s);
  display.display();

  delay(1000);        // delay in between reads for stability
}
```



Temperature measurement

‘Adafruit MCP9808’

- Up to 8 on a single I2C bus with adjustable address pins
- 0.25° C typical precision over -40°C to 125°C range
- 0.0625° C resolution

~700 yen

address select pins

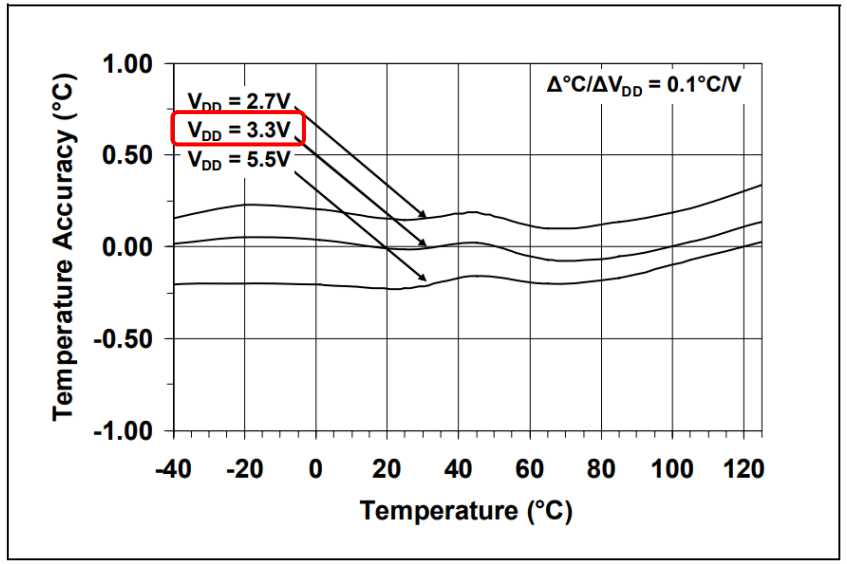
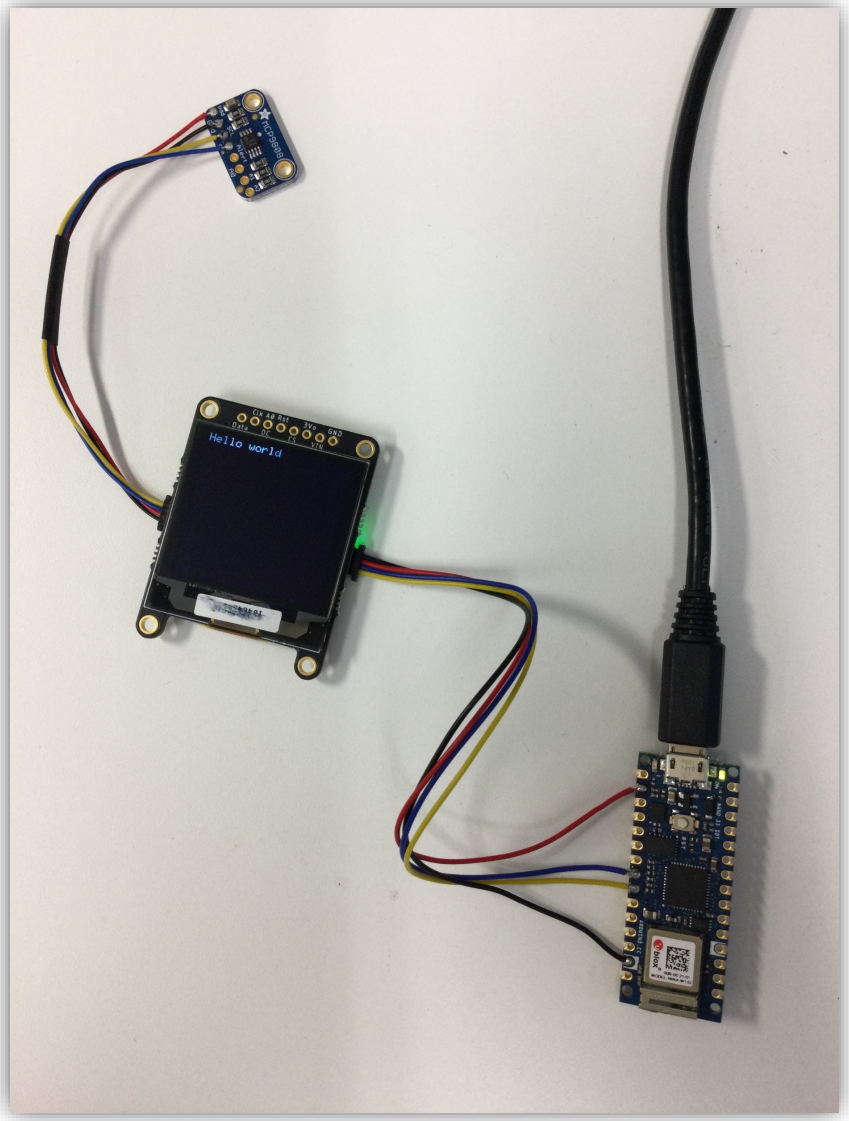


FIGURE 2-11: *Temperature Accuracy vs Supply Voltage.*

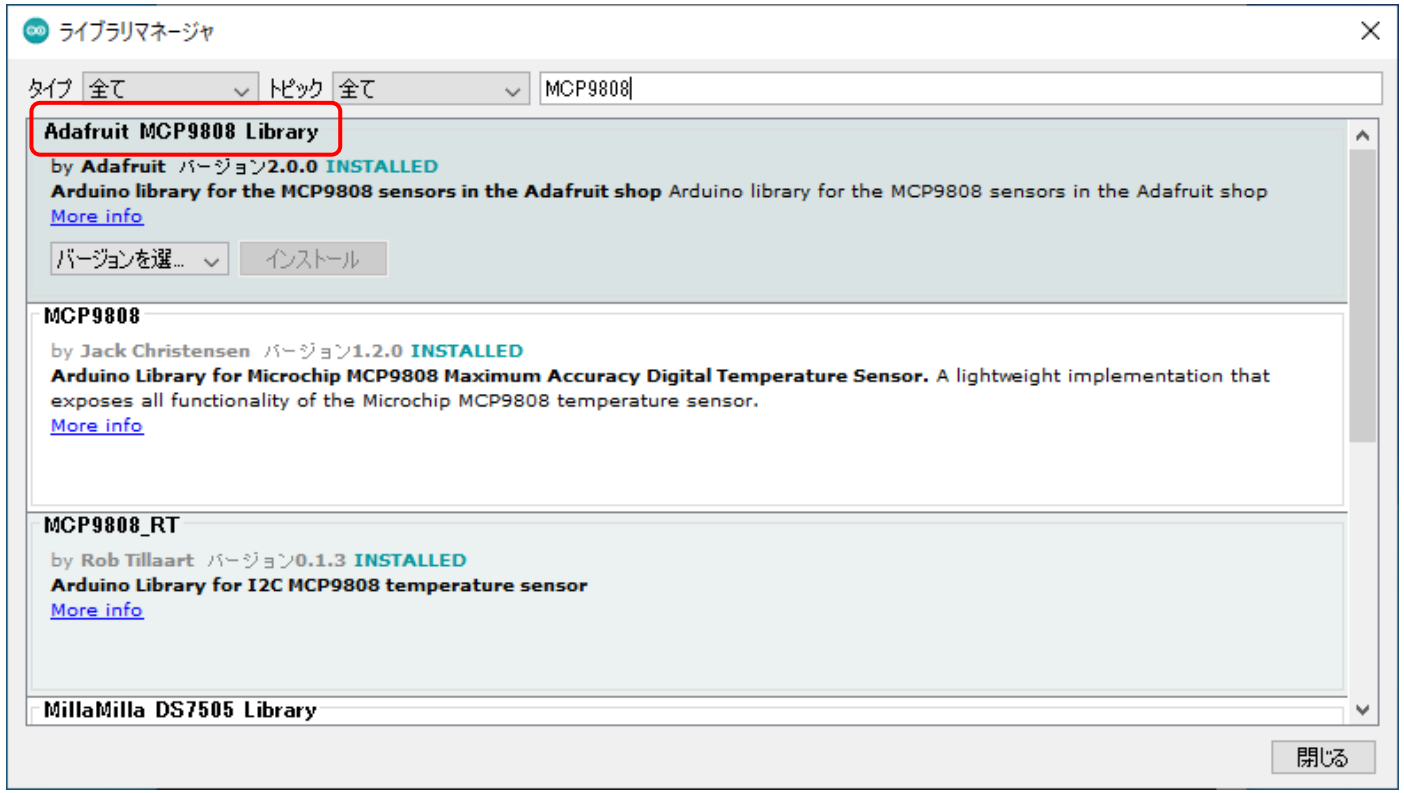
Parameters	Sym	Min	Typ	Max	Unit	Conditions
Temperature Sensor Accuracy						
-20°C < T_A ≤ +100°C	T_{ACY}	-0.5	±0.25	+0.5	°C	V_{DD} = 3.3V
-40°C < T_A ≤ +125°C	T_{ACY}	-1.0	±0.25	+1.0	°C	V_{DD} = 3.3V
Temperature Conversion Time						
0.5°C/bit	t_{CONV}	—	30	—	ms	33s/sec (typical)
0.25°C/bit		—	65	—	ms	15s/sec (typical)
0.125°C/bit		—	130	—	ms	7s/sec (typical)
0.0625°C/bit		—	250	—	ms	4s/sec (typical)

Temperature measurement

- I2C connection

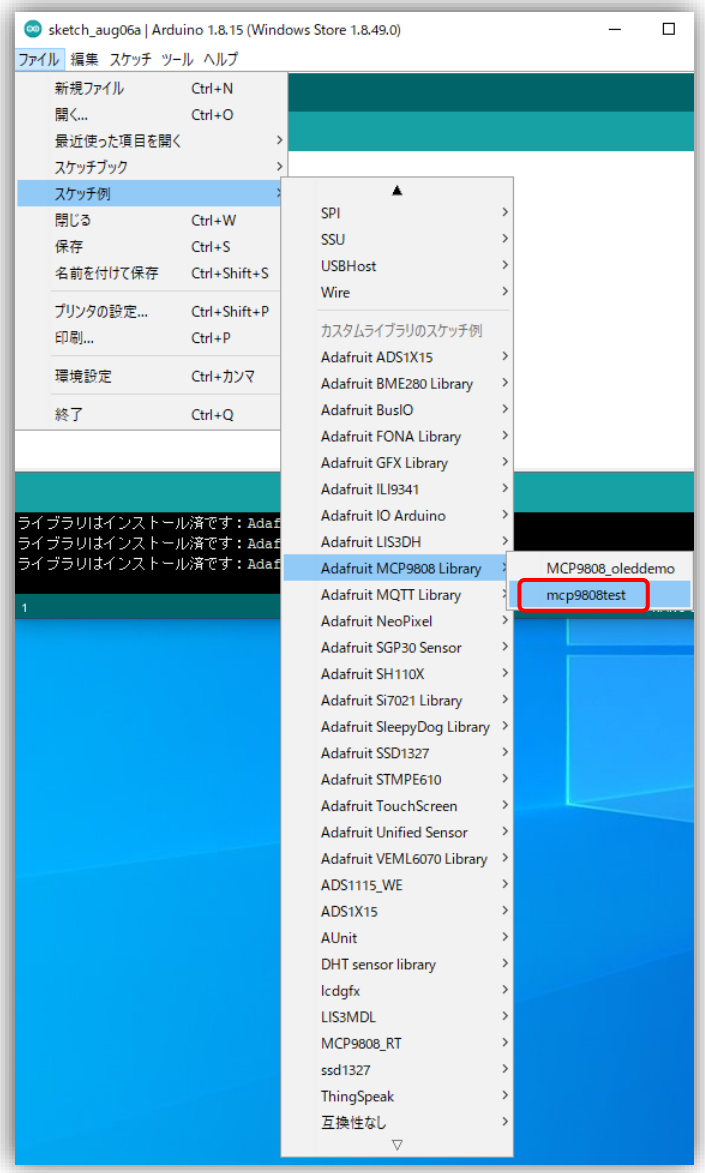


- Install the library of MCP9808



Temperature measurement

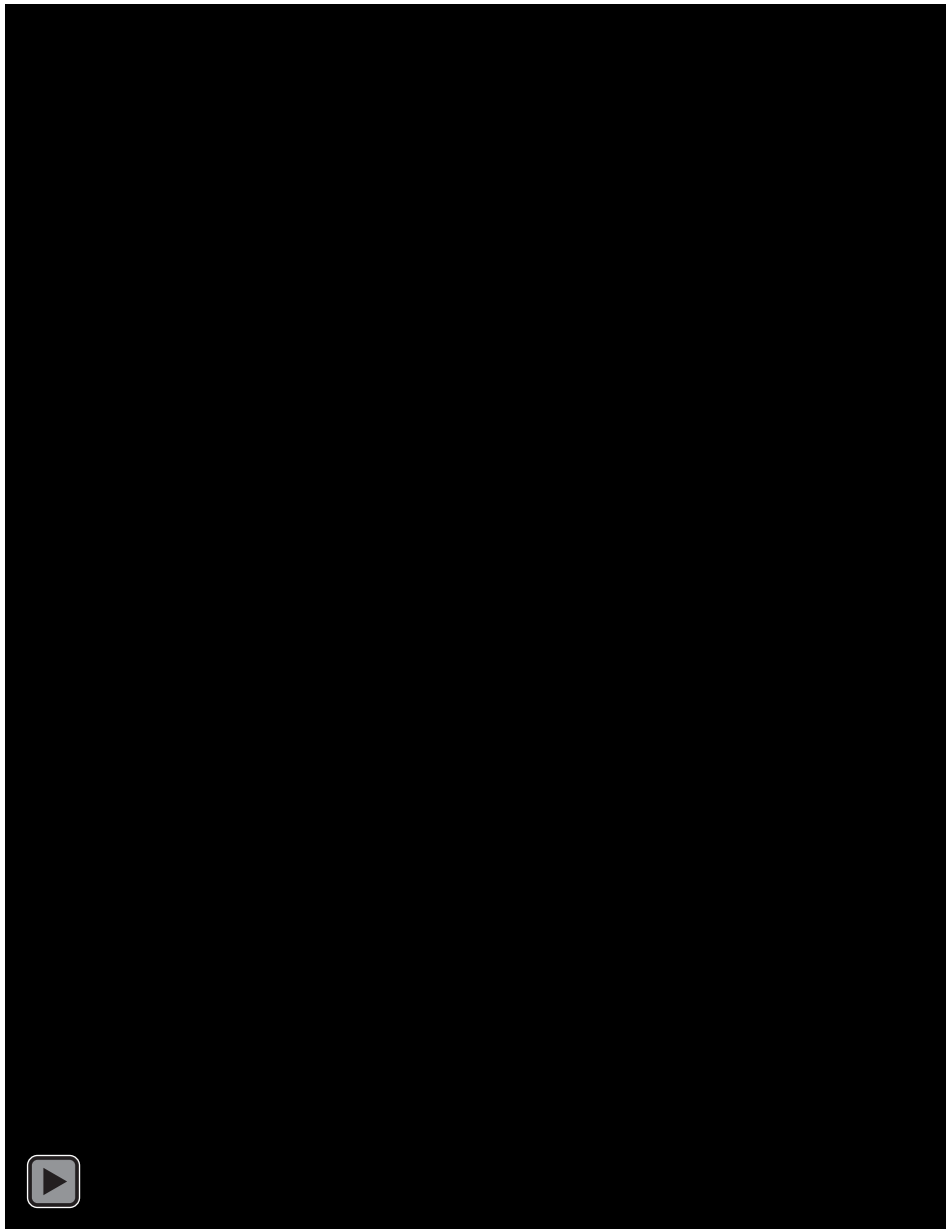
- Open a sample code



- Write it to your Arduino

Temperature measurement

- Test run



Temperature measurement

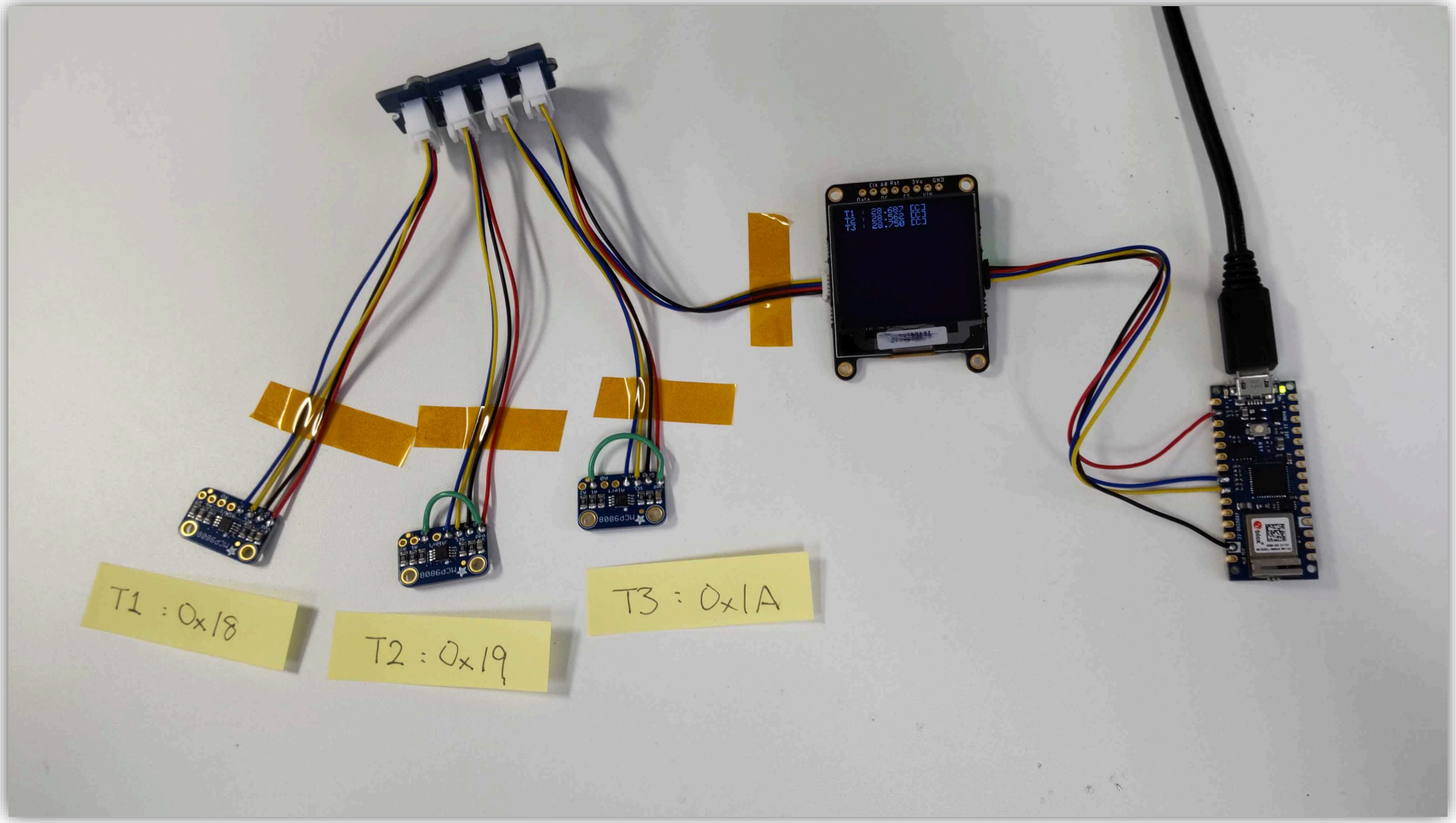
- Cut and paste

Hello_Temperature §

```
#include <Adafruit_SSD1327.h>
#include <Wire.h>
#include "Adafruit_MCP9808.h"

// I2C
#define OLED_RESET -1
Adafruit_SSD1327 display(128, 128, &Wire, OLED_RESET, 1000000);

// Create the MCP9808 temperature sensor object
Adafruit_MCP9808 tempsensor = Adafruit_MCP9808();

// the setup routine runs once when you press reset:
void setup() {
  display.begin(0x3D);
  display.clearDisplay();
  display.display();

  tempsensor.begin(0x18);
  tempsensor.setResolution(3); // sets the resolution mode of reading, the modes are
  // Mode Resolution SampleTime
  //  0    0.5°C       30 ms
  //  1    0.25°C      65 ms
  //  2    0.125°C     130 ms
  //  3    0.0625°C    250 ms
  tempsensor.wake();
}

// the loop routine runs over and over again forever:
void loop() {

  float c = tempsensor.readTempC();

  display.clearDisplay();
  display.setTextSize(1);
  display.setCursor(0, 0);

  display.print("T : "); display.print(c, 3); display.println(" [C]");
  display.display();

  delay(500);        // delay in between reads
}
```

Temperature measurement

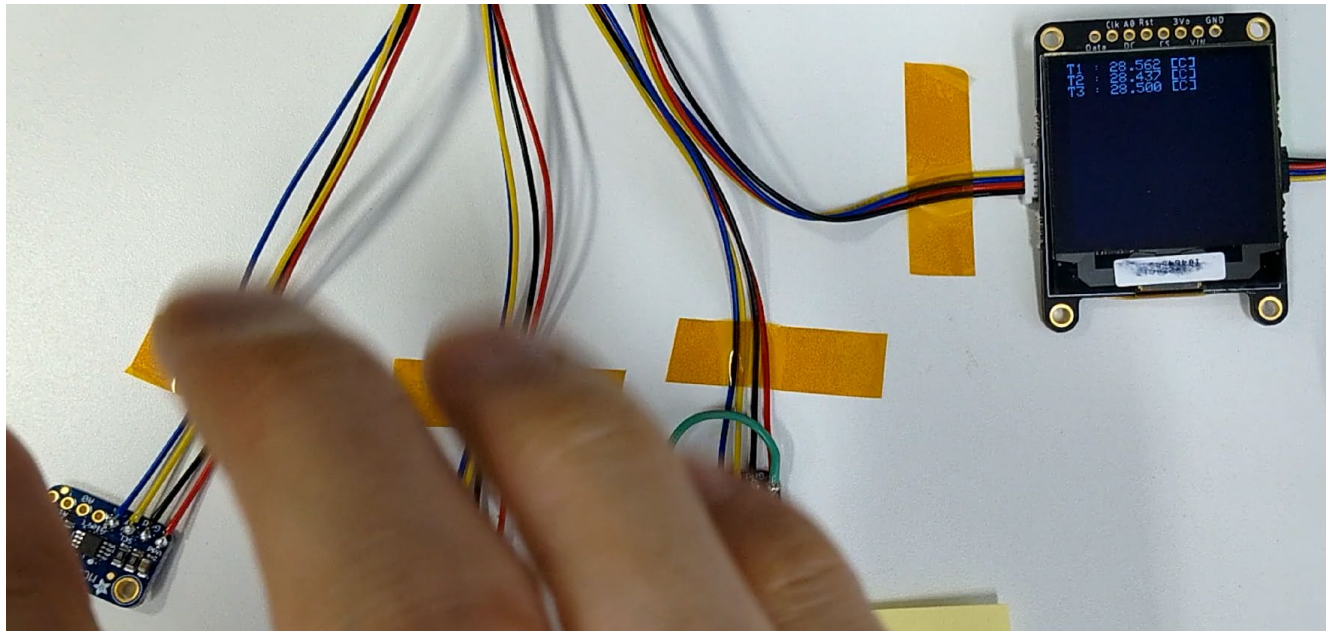
- Multiple sensors



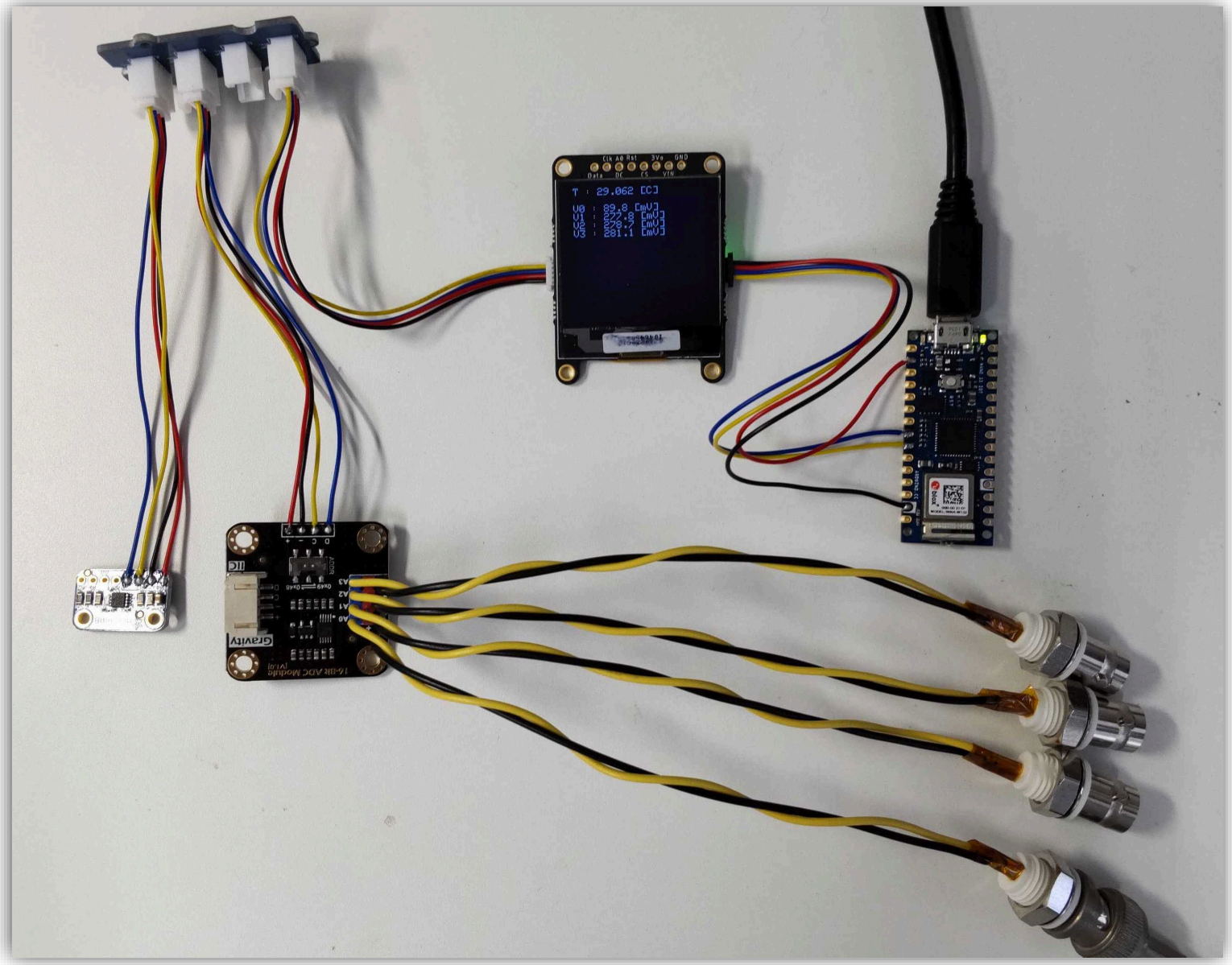
Temperature measurement

- Multiple sensors

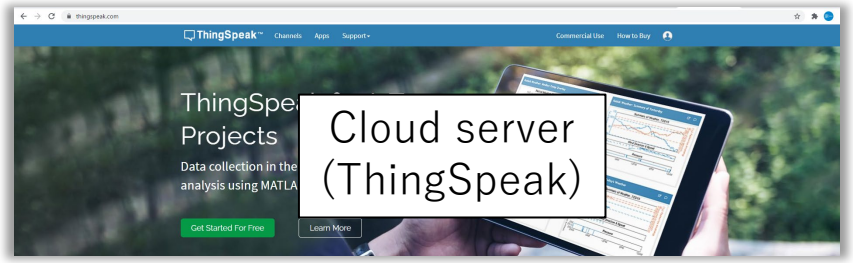
```
#include <Adafruit_SSD1327.h>
#include <Wire.h>
#include "Adafruit_MCP9808.h"

// I2C
#define OLED_RESET -1
Adafruit_SSD1327 display(128, 128, &Wire, OLED_RESET, 1000000);

// Create the MCP9808 temperature sensor object
Adafruit_MCP9808 tempsensor1 = Adafruit_MCP9808();
Adafruit_MCP9808 tempsensor2 = Adafruit_MCP9808();
Adafruit_MCP9808 tempsensor3 = Adafruit_MCP9808();

// the setup routine runs once when you press reset:
void setup() {
  display.begin(0x3D);
  display.clearDisplay();
  display.display();

  tempsensor1.begin(0x18);
  tempsensor2.begin(0x19);
  tempsensor3.begin(0x1A);

  tempsensor1.setResolution(3); // sets the resolution mode of reading, the modes are defined in tl
  tempsensor2.setResolution(3); // sets the resolution mode of reading, the modes are defined in tl
  tempsensor3.setResolution(3); // sets the resolution mode of reading, the modes are defined in tl
  // Mode Resolution SampleTime
  //  0    0.5°C       30 ms
  //  1    0.25°C      65 ms
  //  2    0.125°C     130 ms
  //  3    0.0625°C    250 ms
  tempsensor1.wake();
  tempsensor2.wake();
  tempsensor3.wake();
}

// the loop routine runs over and over again forever:
void loop() {

  float c1 = tempsensor1.readTempC();
  float c2 = tempsensor2.readTempC();
  float c3 = tempsensor3.readTempC();

  display.clearDisplay();
  display.setTextSize(1);
  display.setCursor(0, 0);

  display.print("T1 : "); display.print(c1, 3); display.println(" [C]");
  display.print("T2 : "); display.print(c2, 3); display.println(" [C]");
  display.print("T3 : "); display.print(c3, 3); display.println(" [C]");
  display.display();

  delay(500);        // delay in between reads
}
```



We can add an ADC device as well



Today's goal

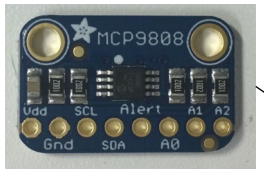


Internet

Internet

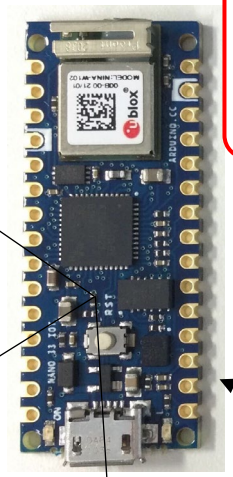
Inside your laboratory

Outside your laboratory



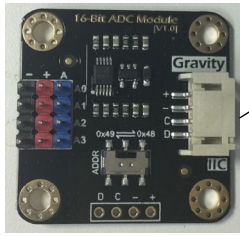
Temperature measurement

I2C



Wi-Fi

Wi-Fi router



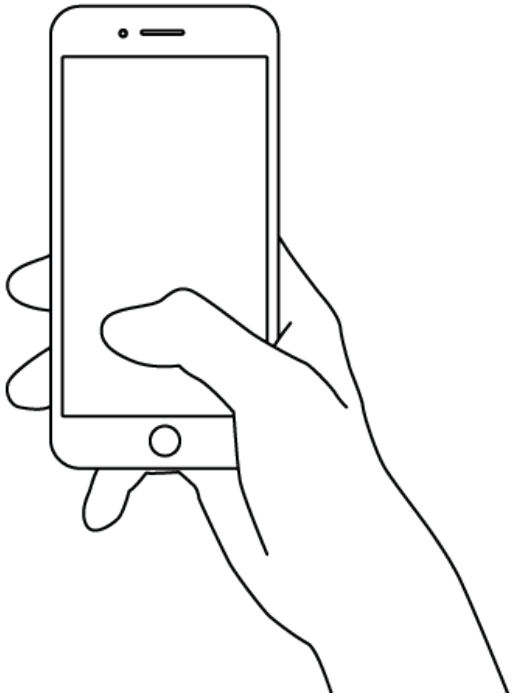
I2C

Voltage measurement
(laser power, MOT fluorescence)

I2C

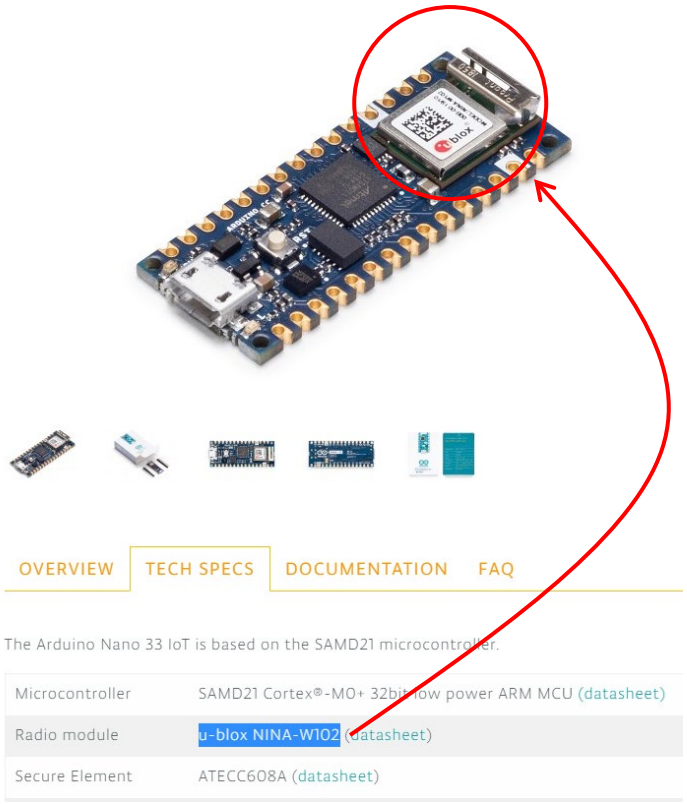
External trigger for upload
data to the cloud

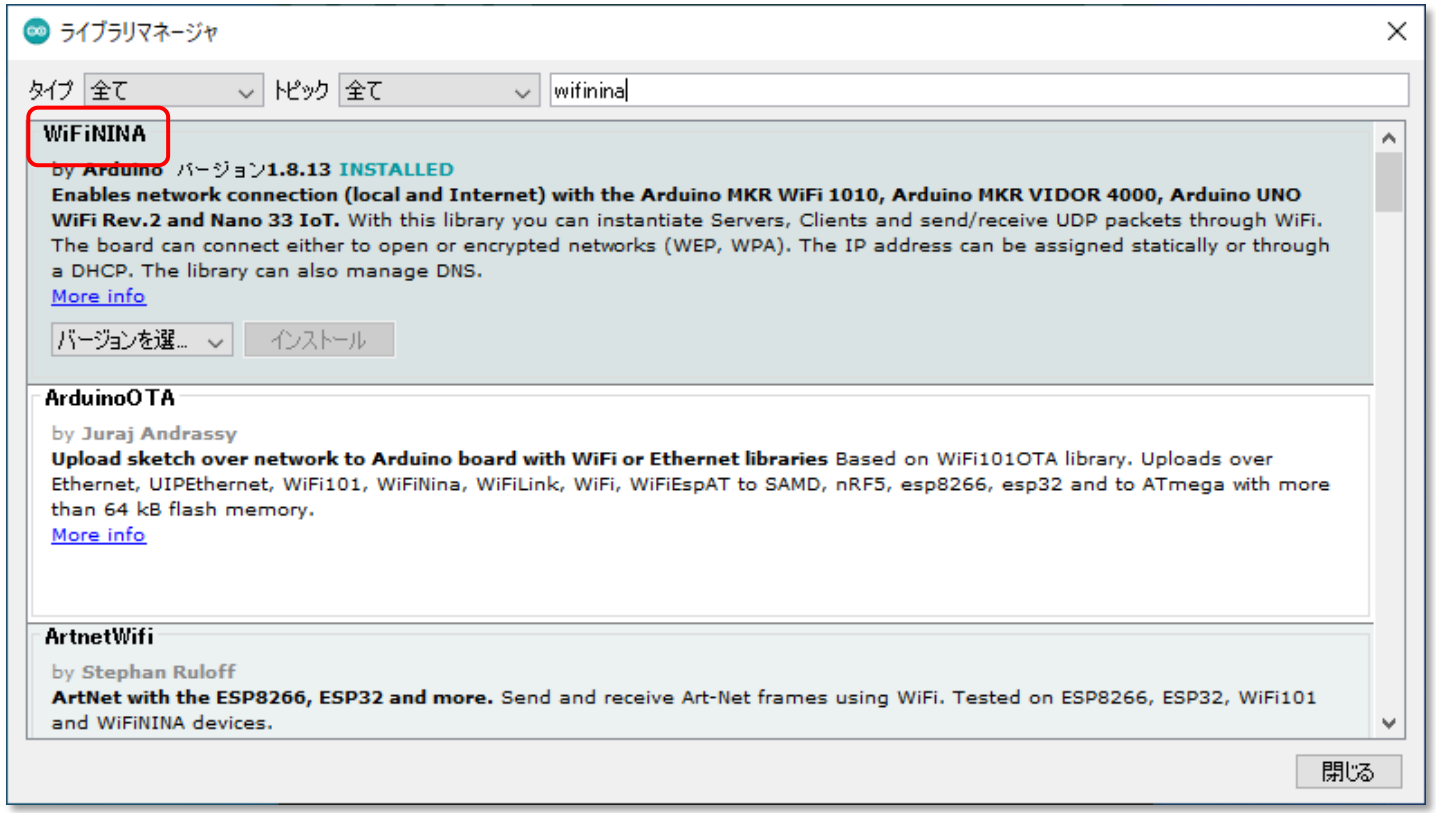
Displaying experimental
conditions on an OLED



Wi-Fi connection

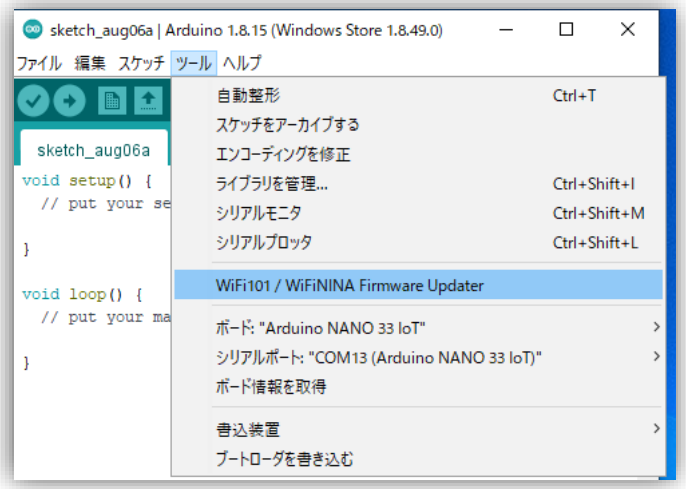
- Install the library of WiFiNINA

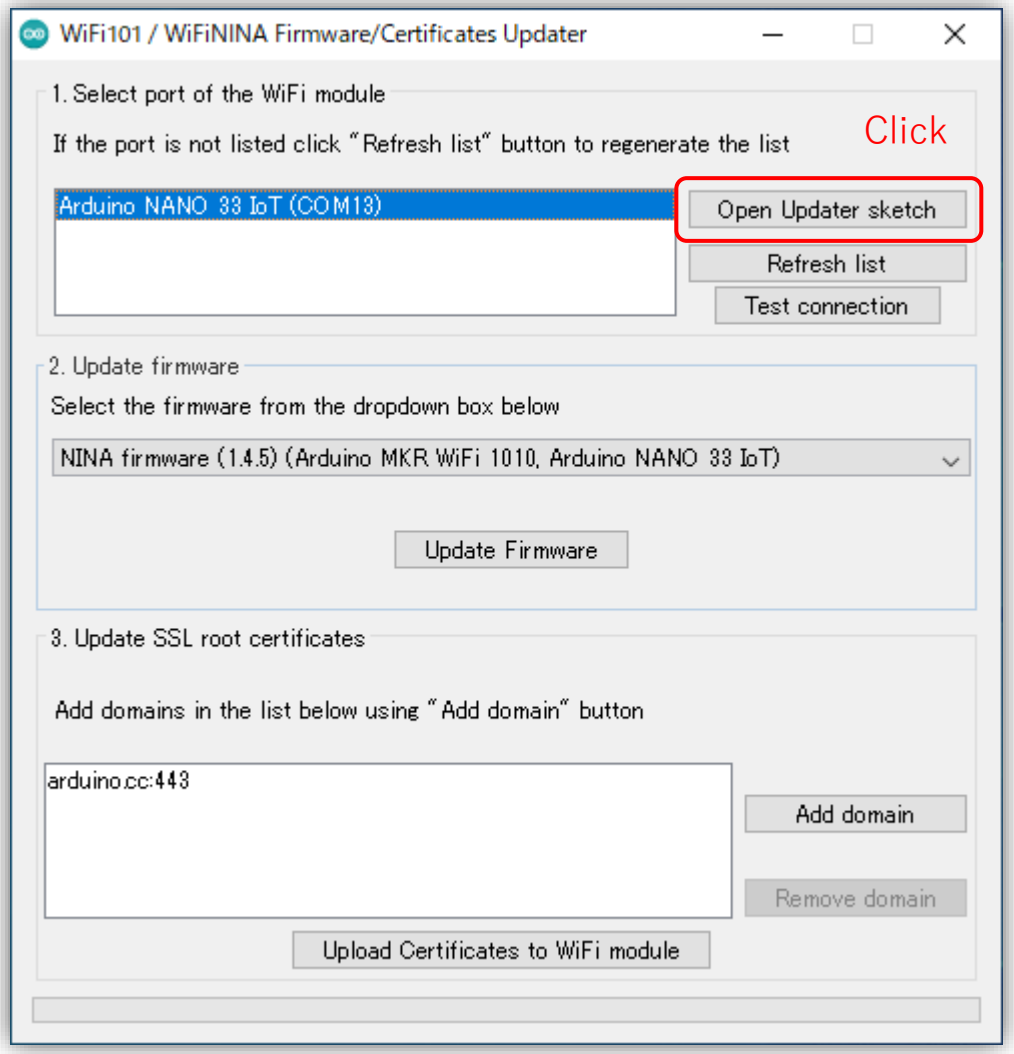




Wi-Fi connection

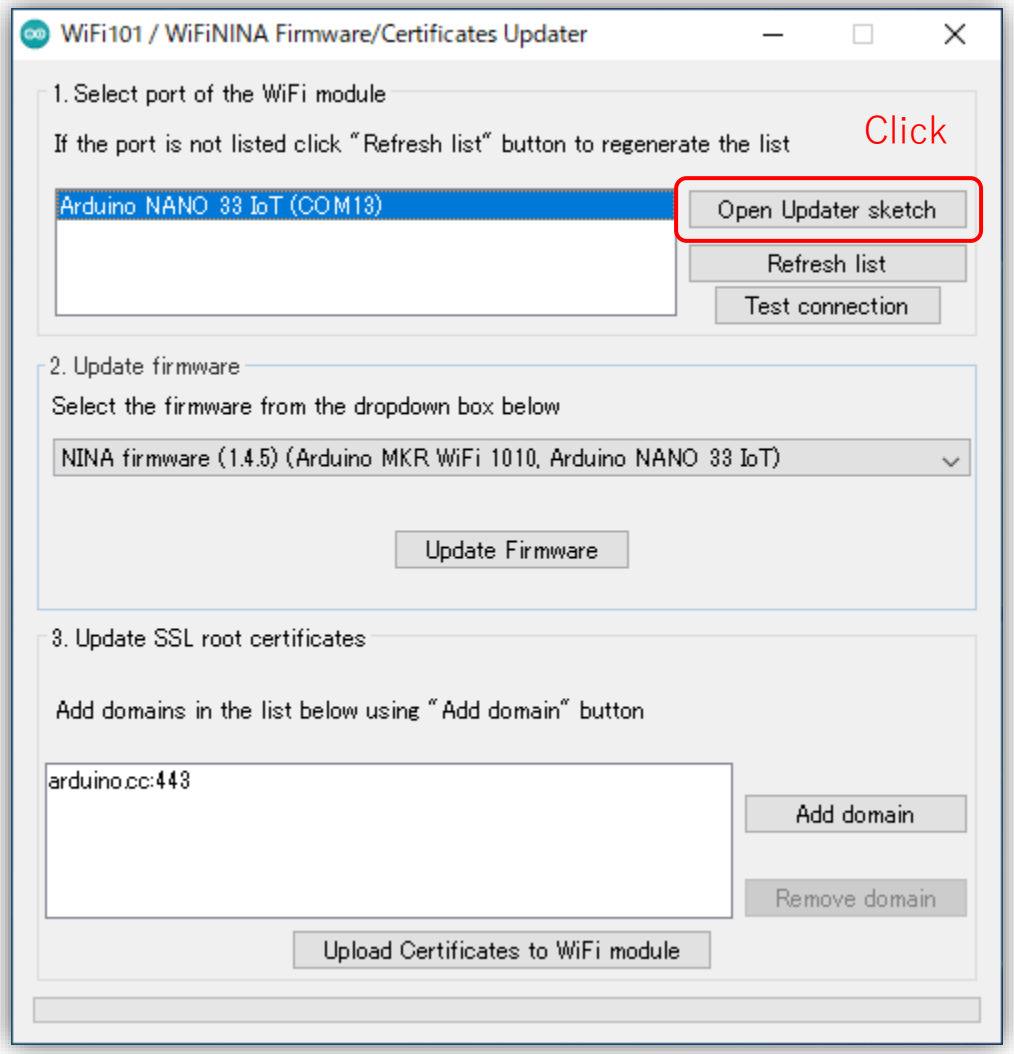
- Update the firmware

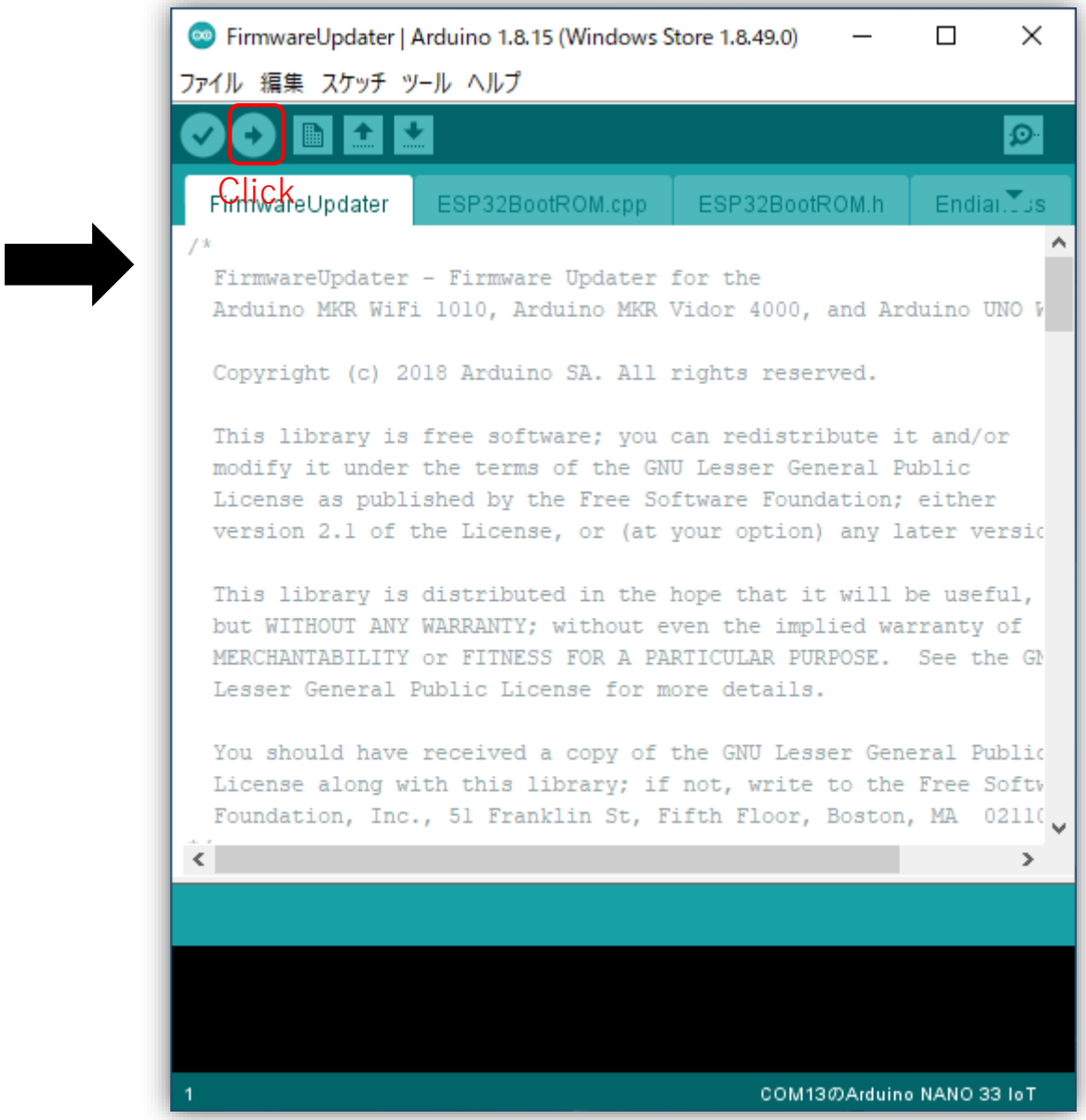




Wi-Fi connection

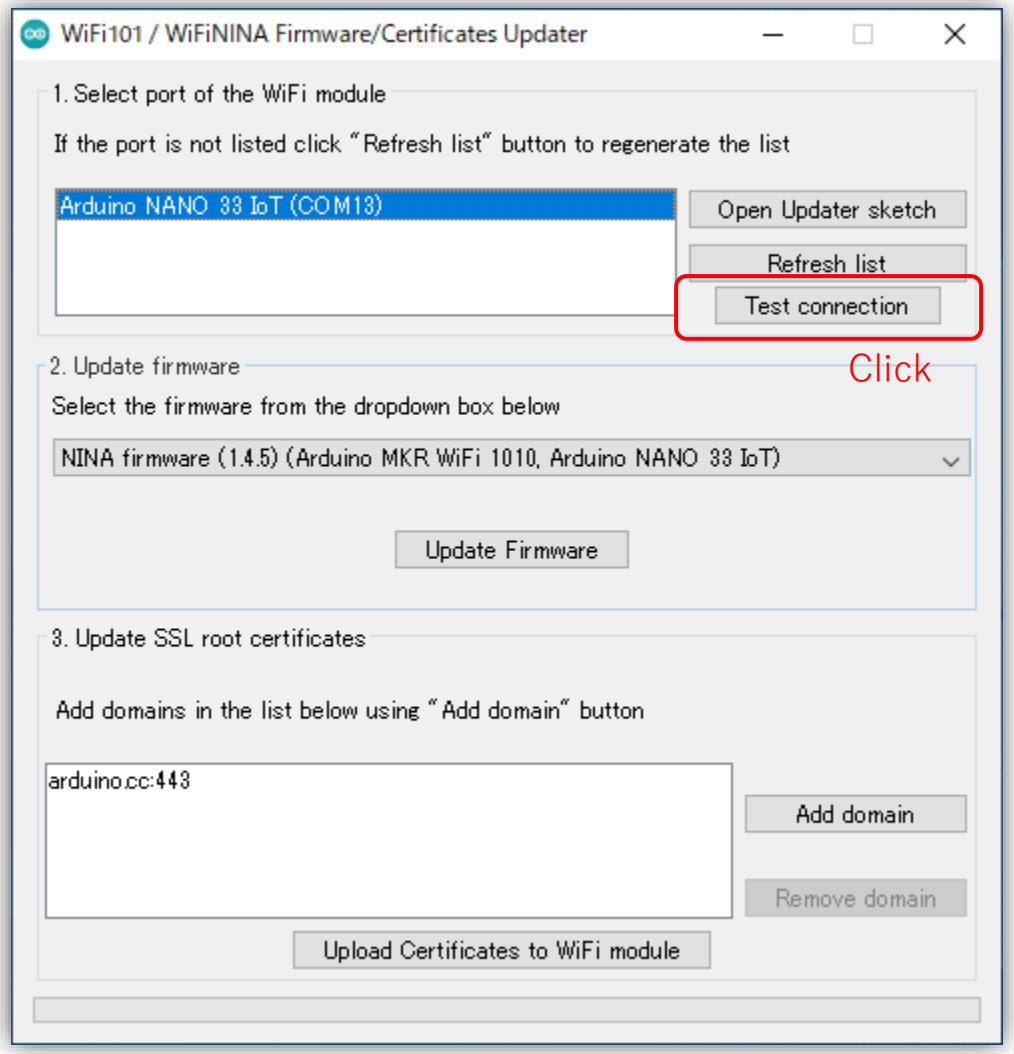
- Update the firmware

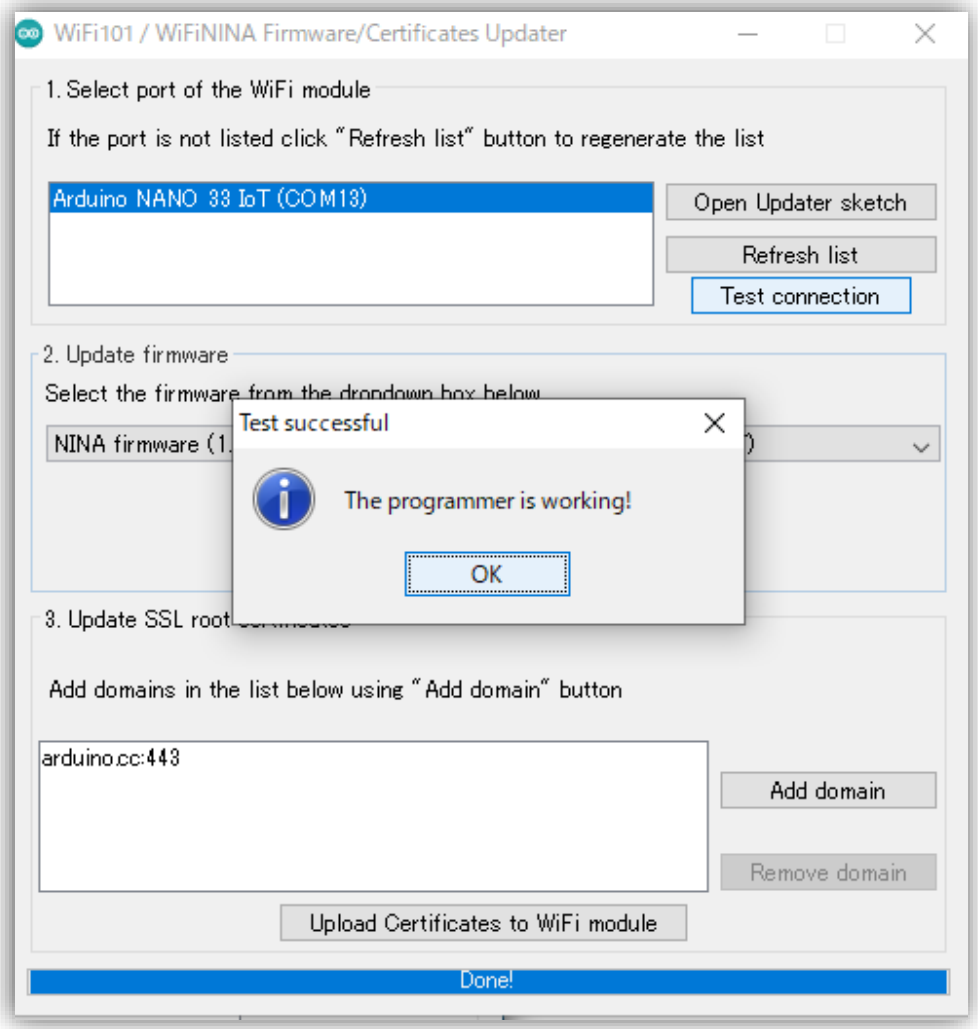




Wi-Fi connection

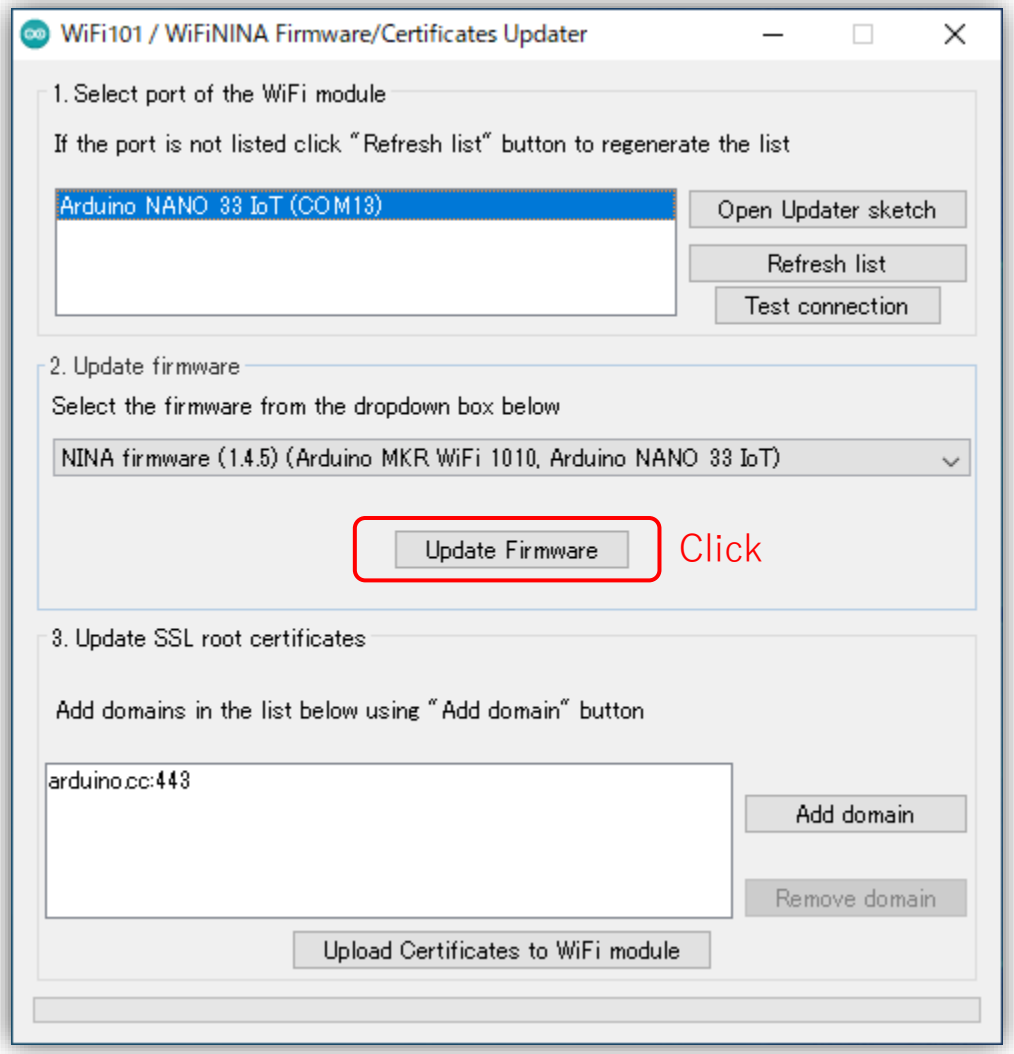
- Update the firmware

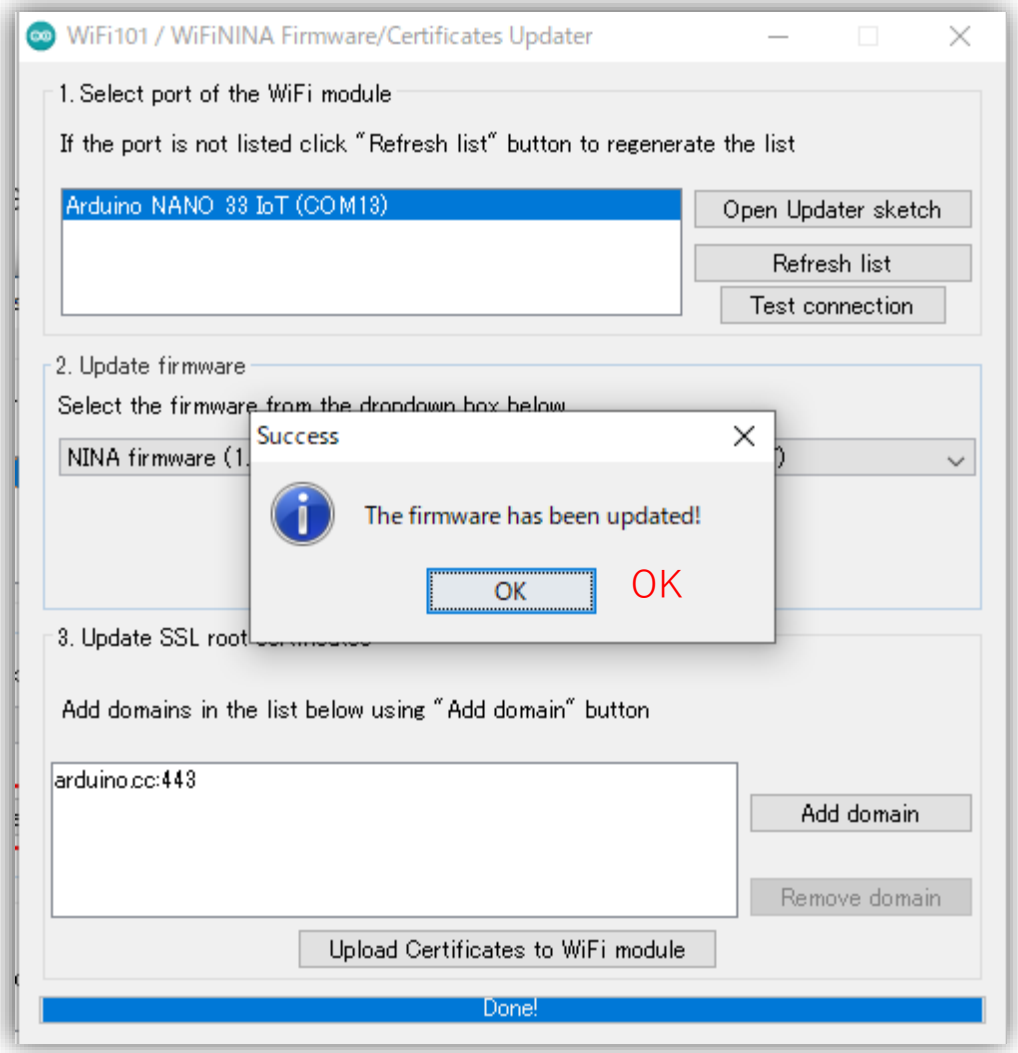




Wi-Fi connection

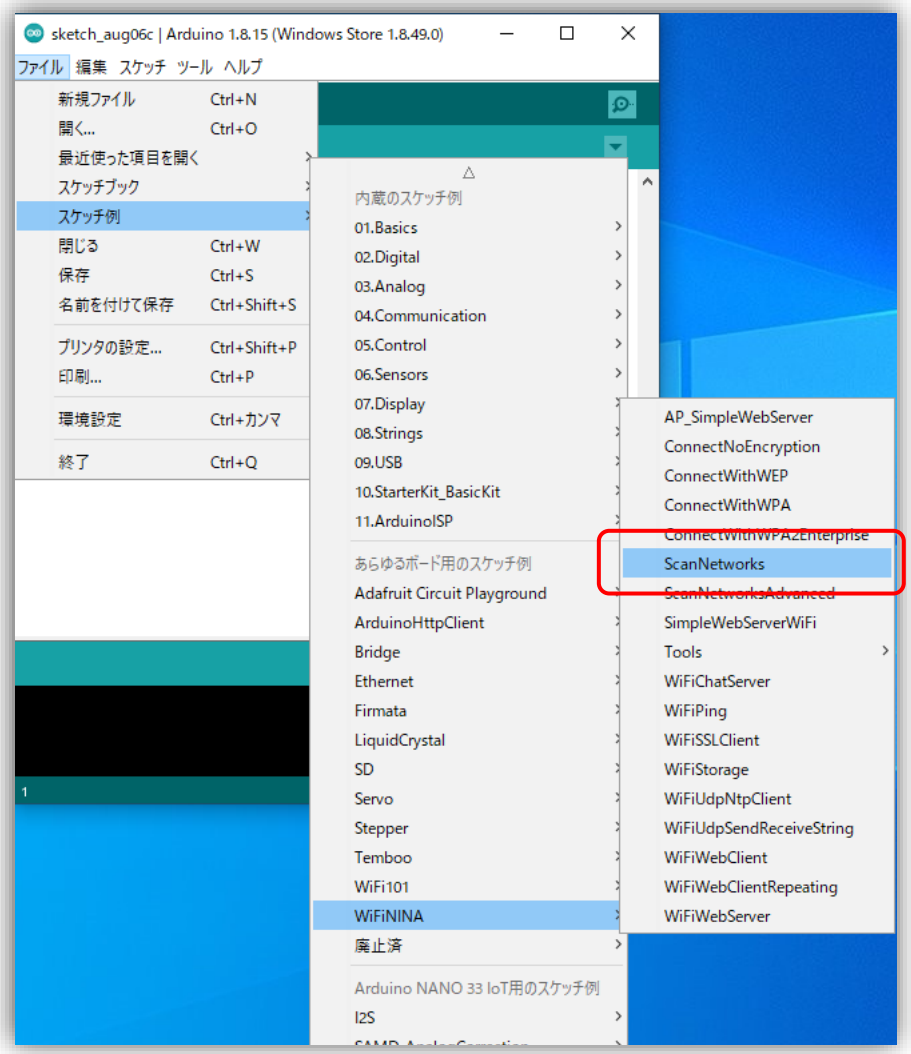
- Update the firmware

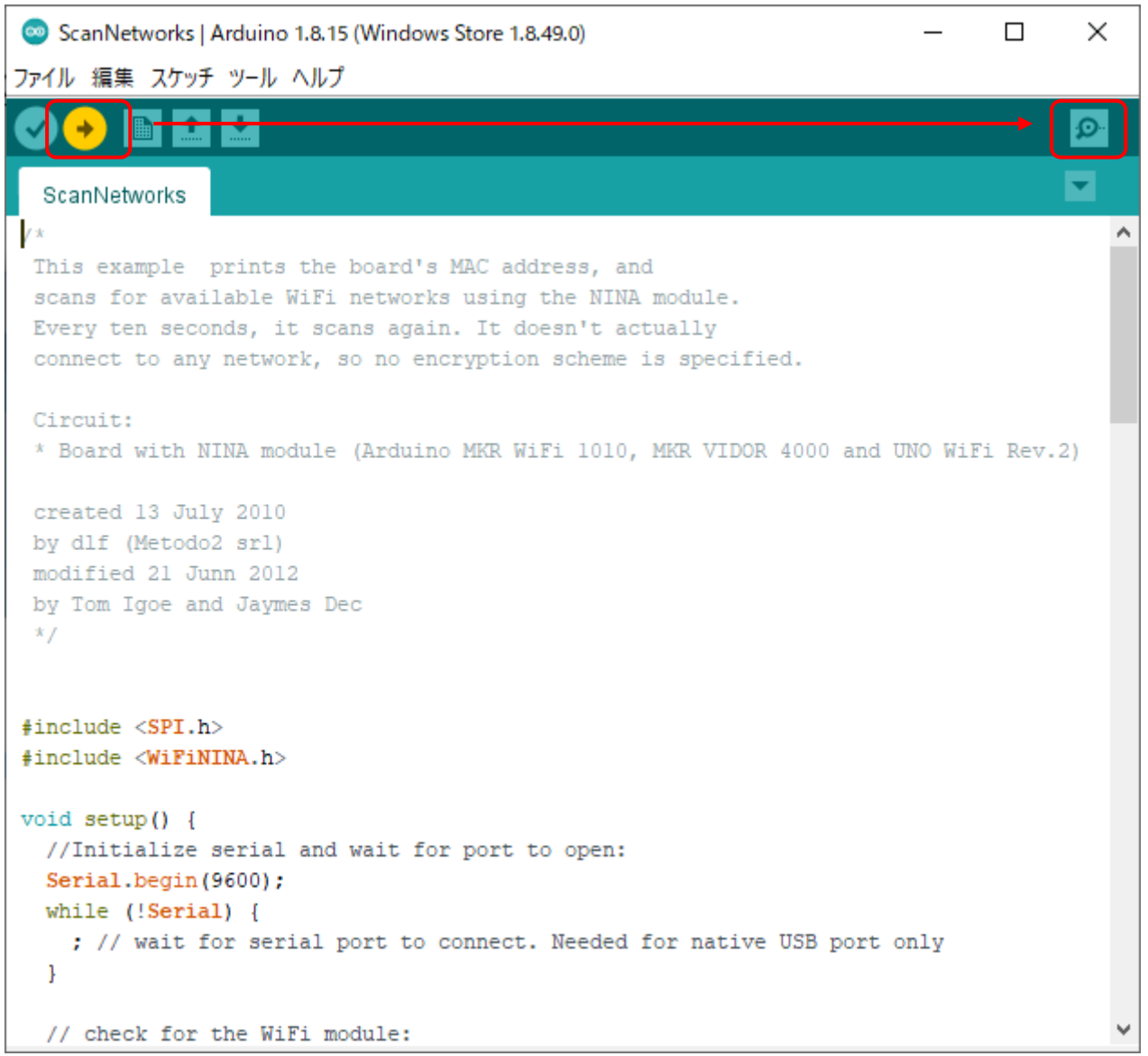




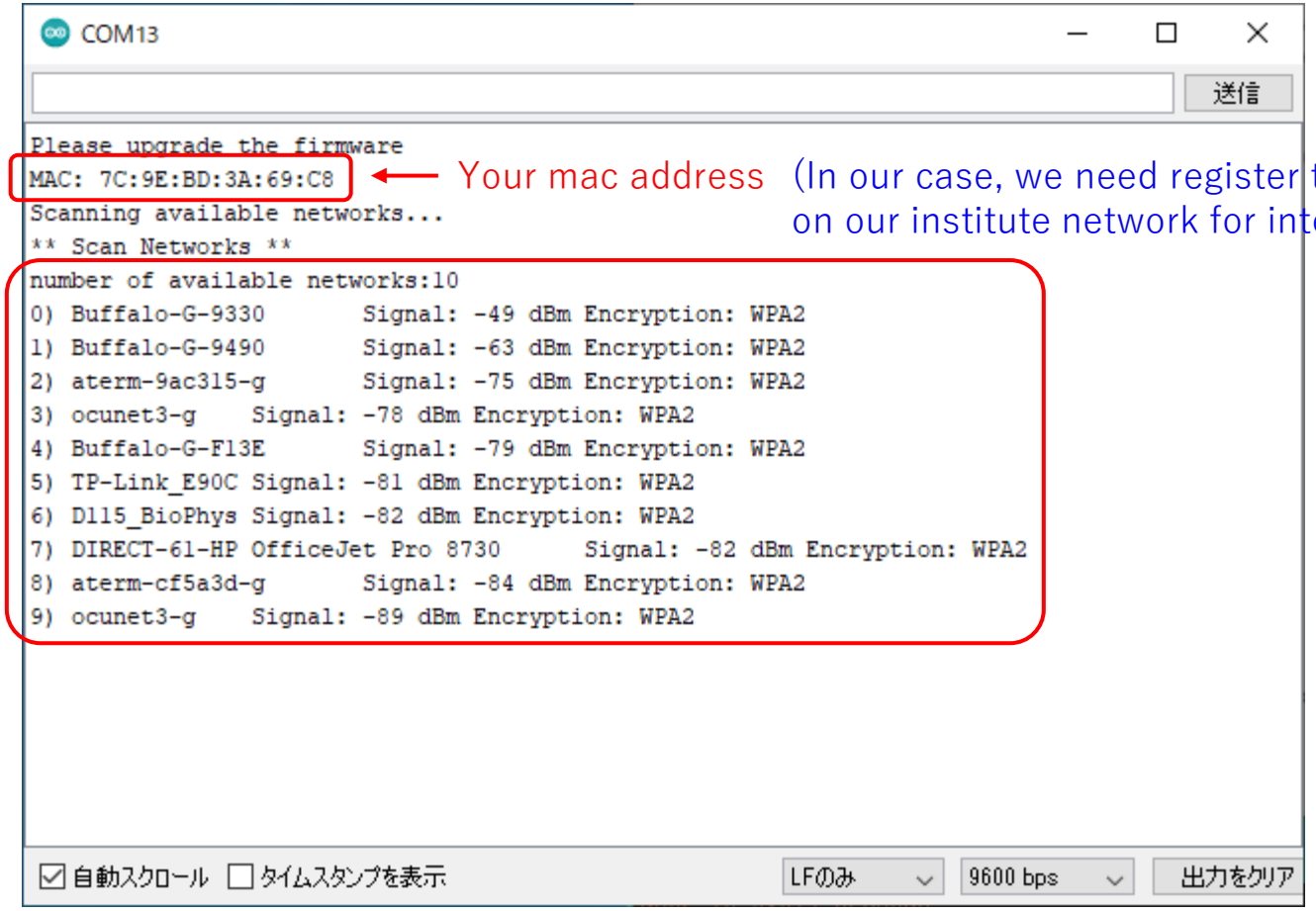
Wi-Fi connection : How to check the mac address

- Open a sample code





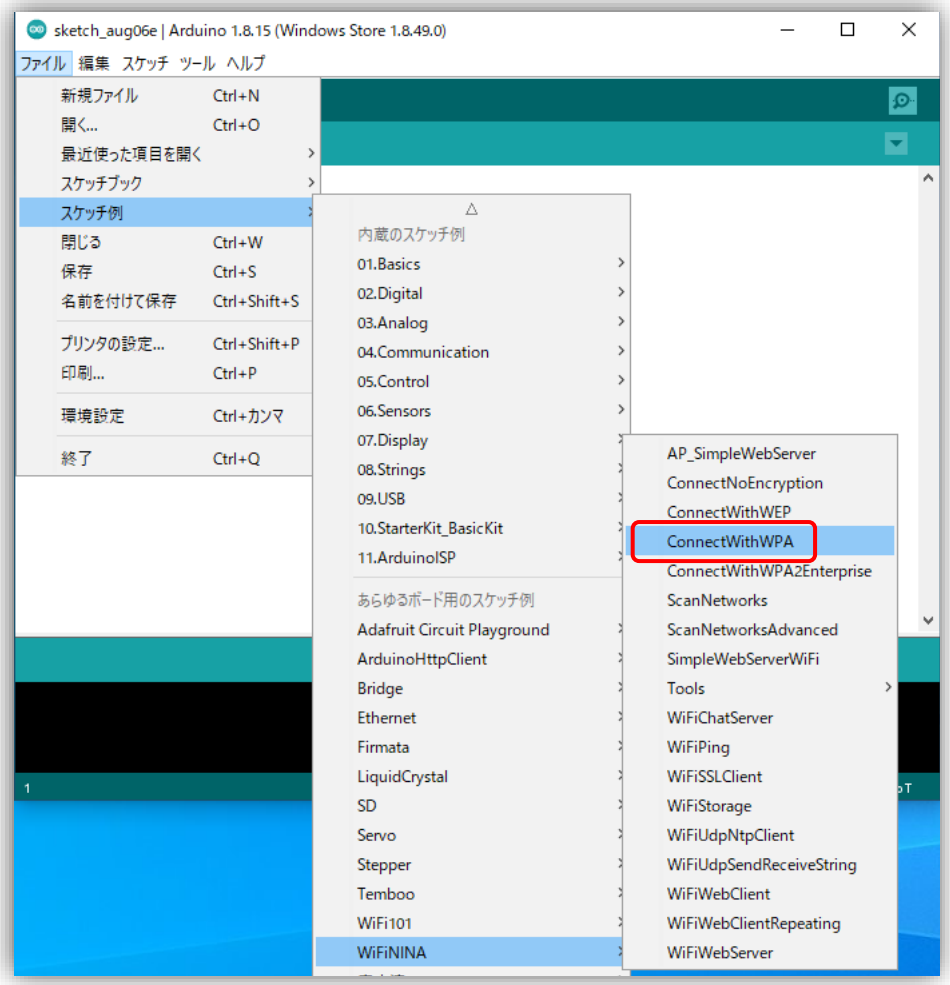
Wi-Fi connection : How to check the mac address

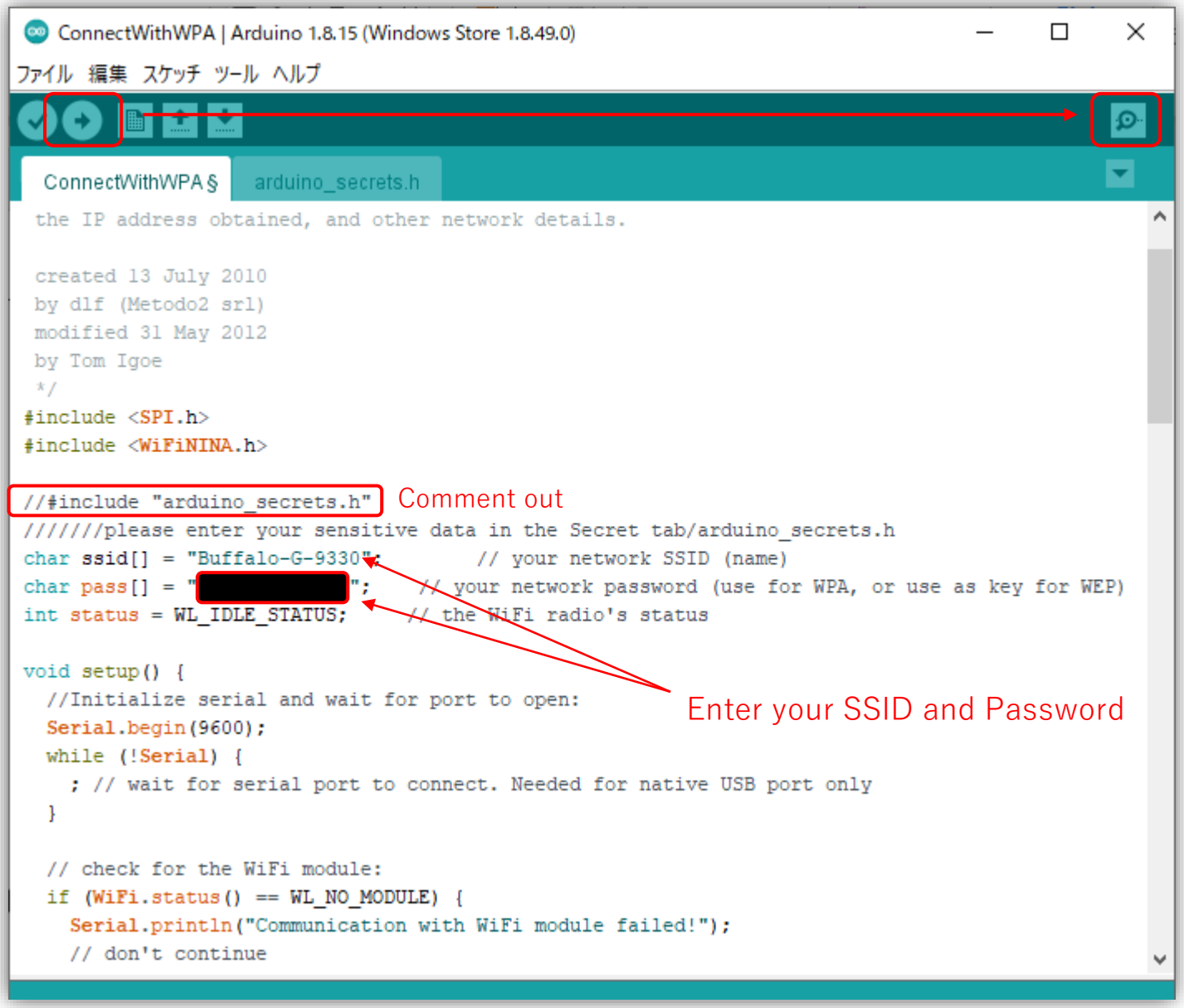


Your mac address (In our case, we need register the mac address on our institute network for internet connection)

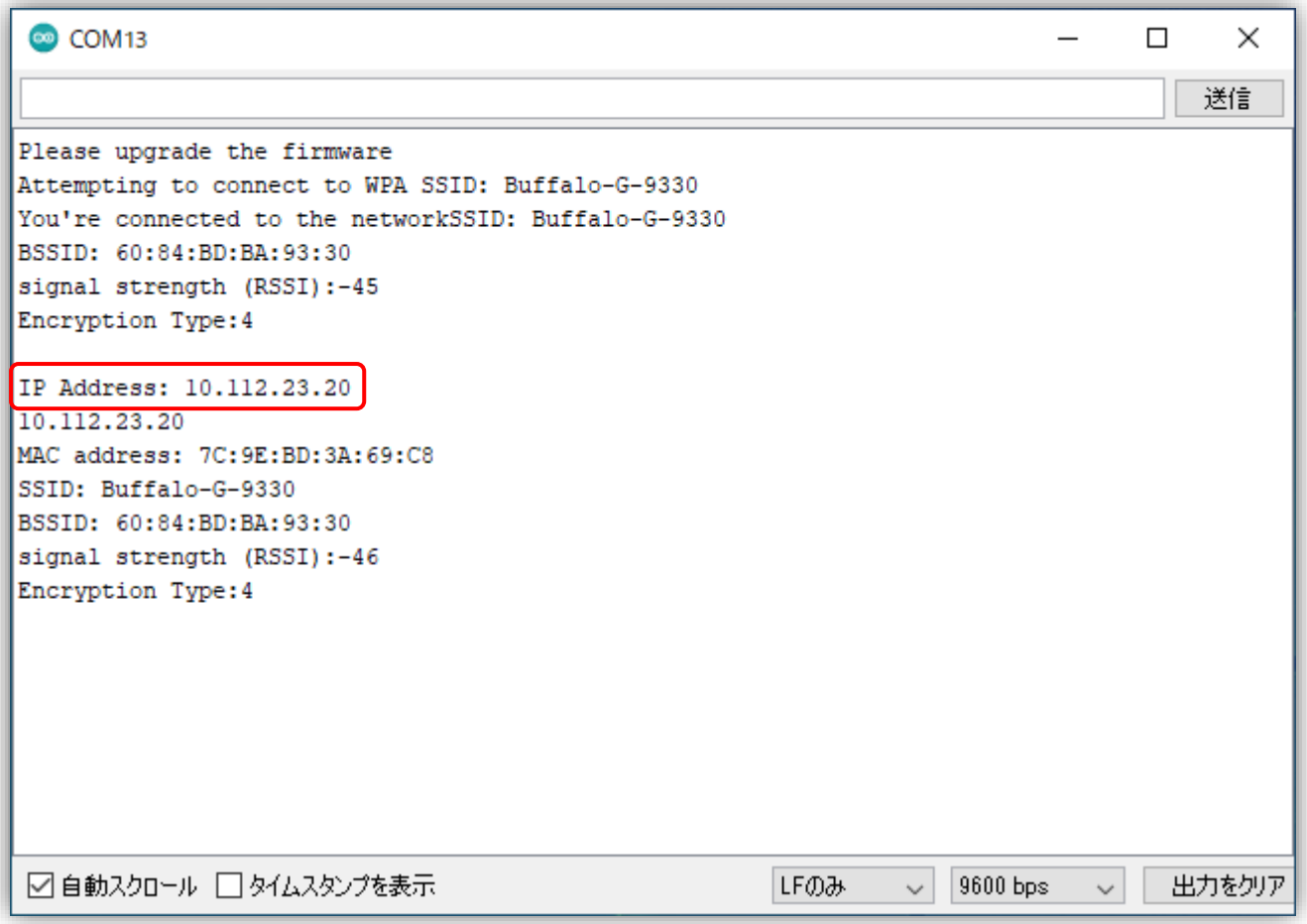
Wi-Fi connection : How to check the mac address

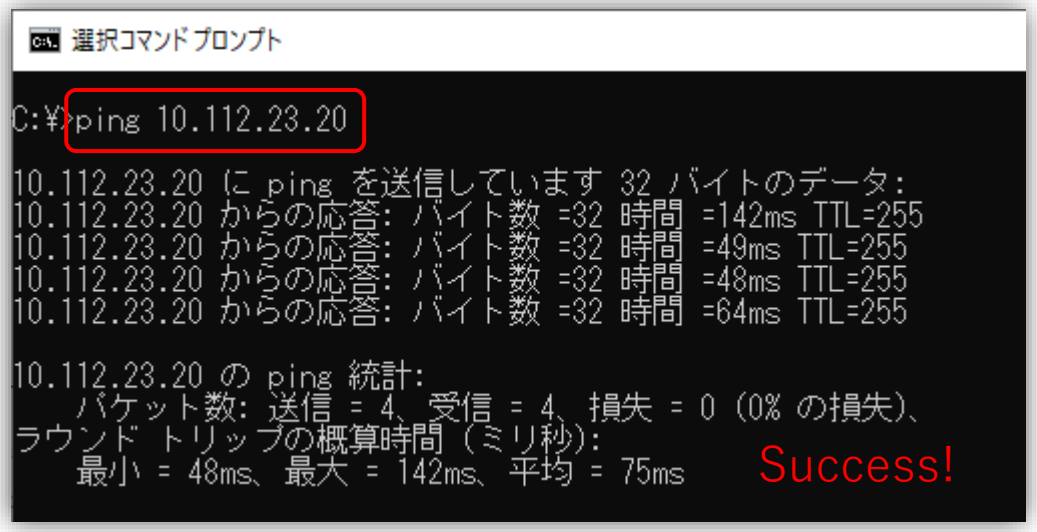
- Connect to the internet





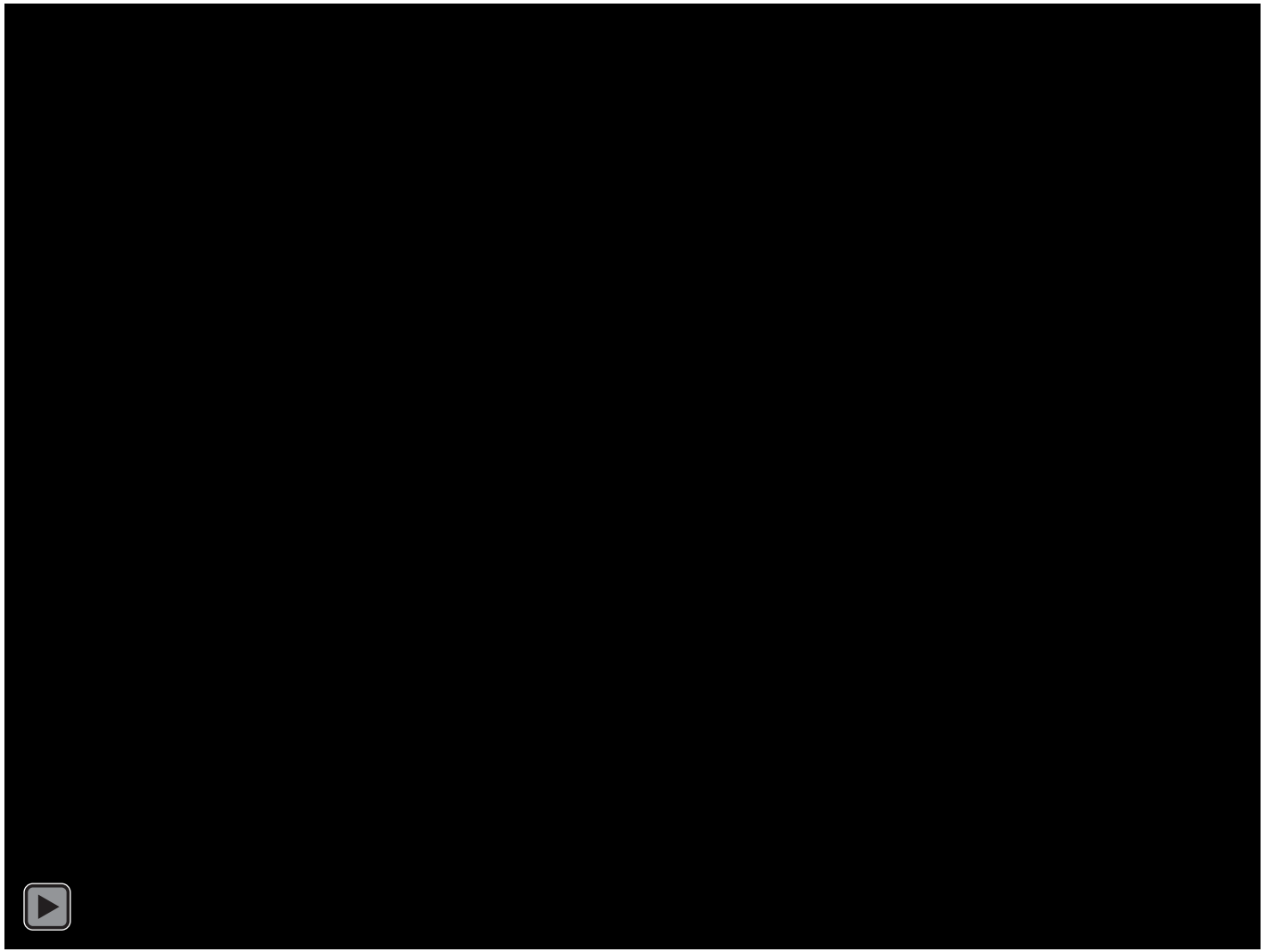
Wi-Fi connection : How to confirm the connection



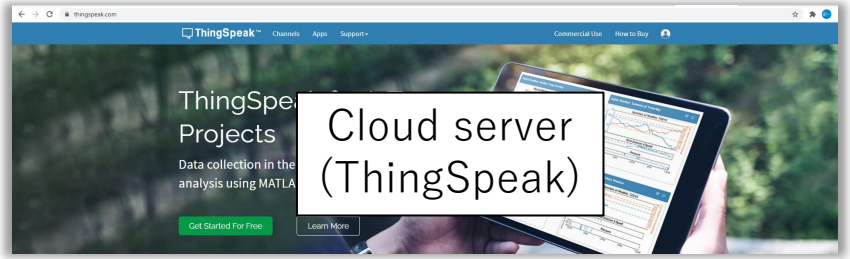


Wi-Fi connection : Automatic internet connection in standalone operating

- Cut and paste, then edit



Today's goal



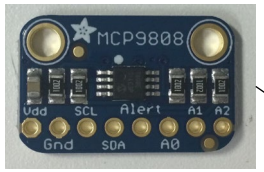
Cloud server (ThingSpeak)

Internet

Outside your laboratory

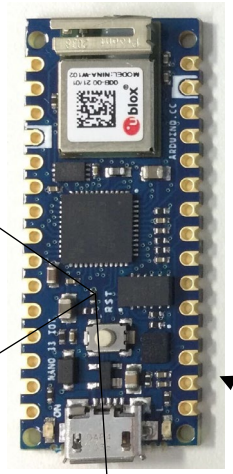
Inside your laboratory

Internet



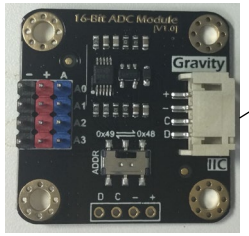
Temperature measurement

I2C



Wi-Fi

Wi-Fi router



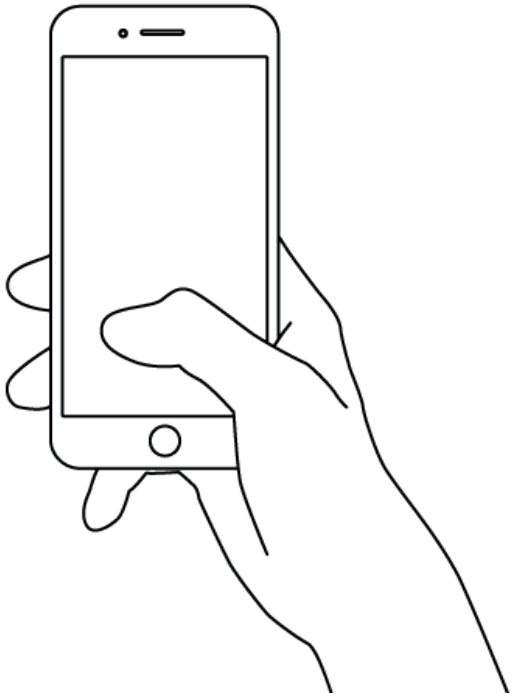
I2C

Voltage measurement
(laser power, MOT fluorescence)

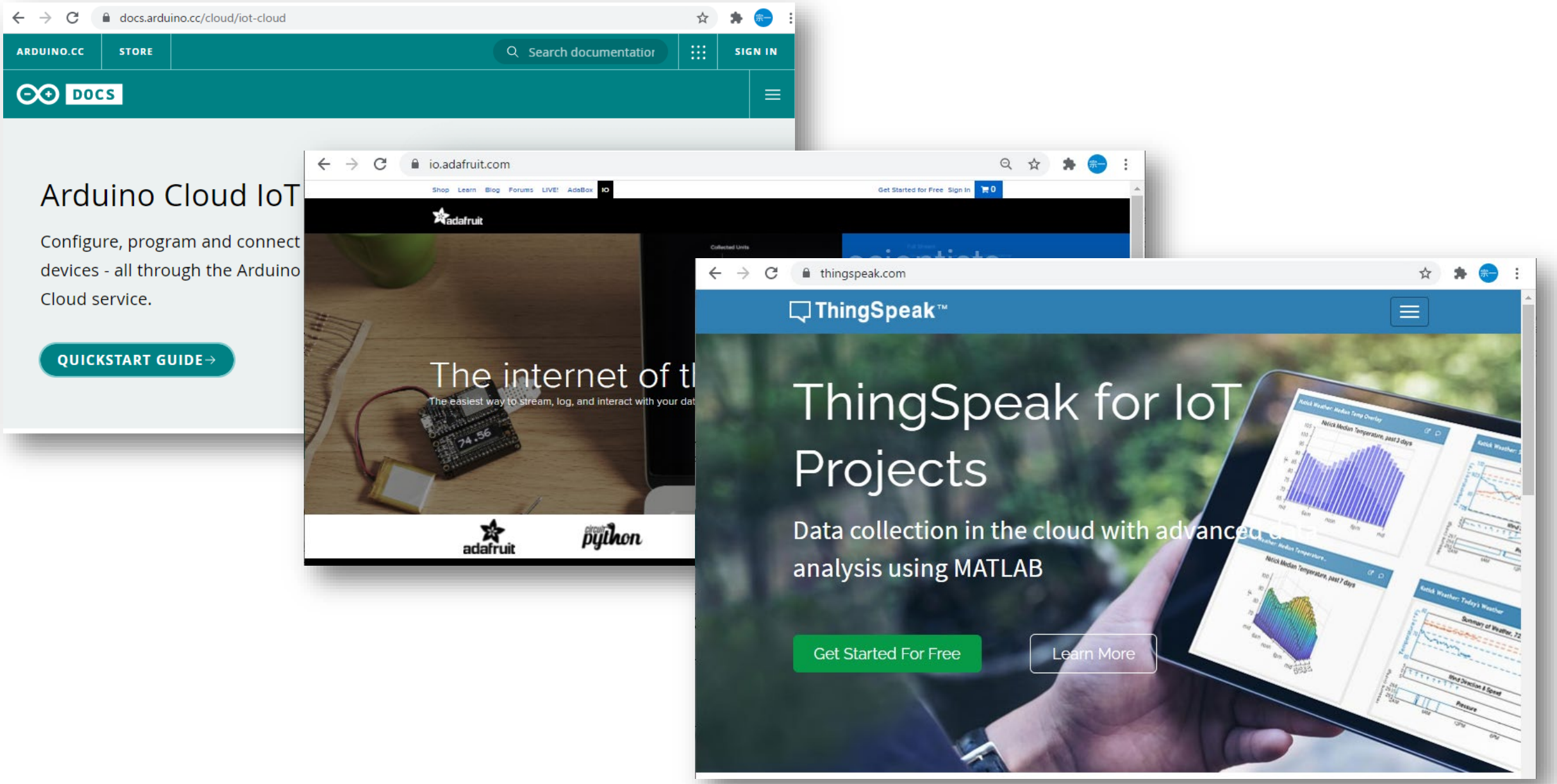
I2C

External trigger for upload data to the cloud

Displaying experimental conditions on an OLED

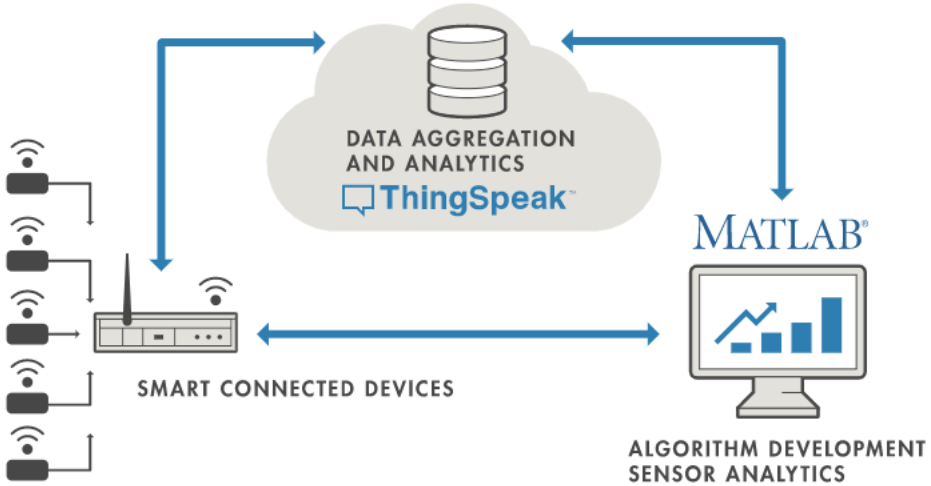


Upload data to an IOT cloud server



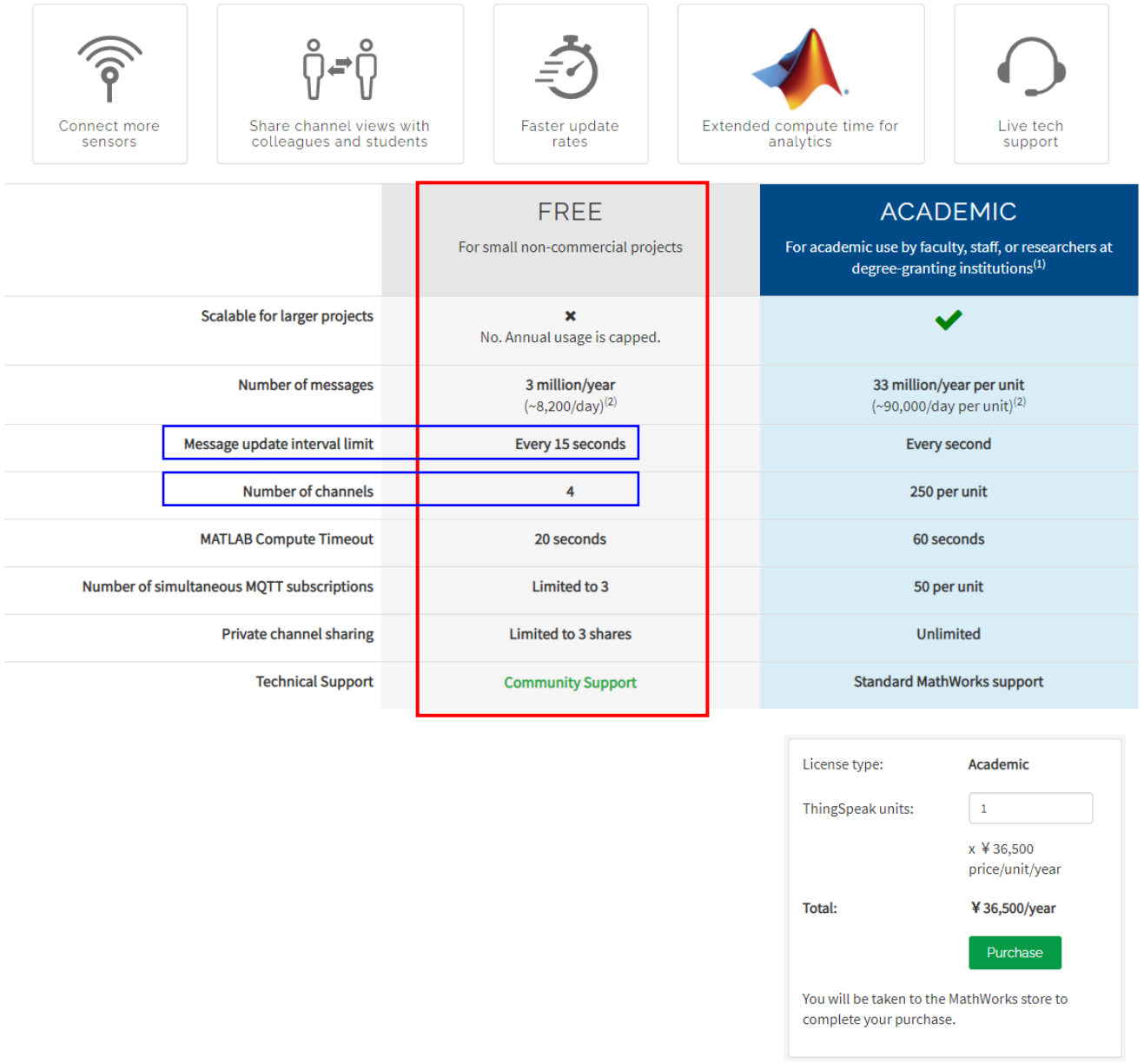
Upload data to an IOT cloud server

https://thingspeak.com/



https://thingspeak.com/pages/learn_more

Upload data to an IOT cloud server



Connect more sensors

Share channel views with colleagues and students

Faster update rates

Extended compute time for analytics

Live tech support

	FREE For small non-commercial projects	ACADEMIC For academic use by faculty, staff, or researchers at degree-granting institutions(1)
Scalable for larger projects	✖ No. Annual usage is capped.	✔
Number of messages	3 million/year (~8,200/day)(2)	33 million/year per unit (~90,000/day per unit)(2)
Message update interval limit	Every 15 seconds	Every second
Number of channels	4	250 per unit
MATLAB Compute Timeout	20 seconds	60 seconds
Number of simultaneous MQTT subscriptions	Limited to 3	50 per unit
Private channel sharing	Limited to 3 shares	Unlimited
Technical Support	Community Support	Standard MathWorks support

License type: Academic

ThingSpeak units: 1

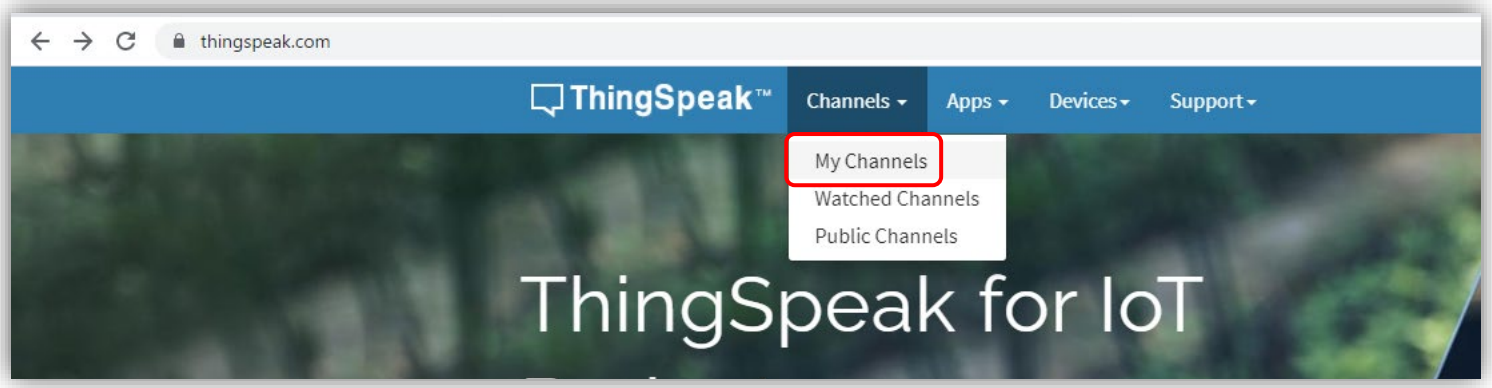
x ¥ 36,500 price/unit/year

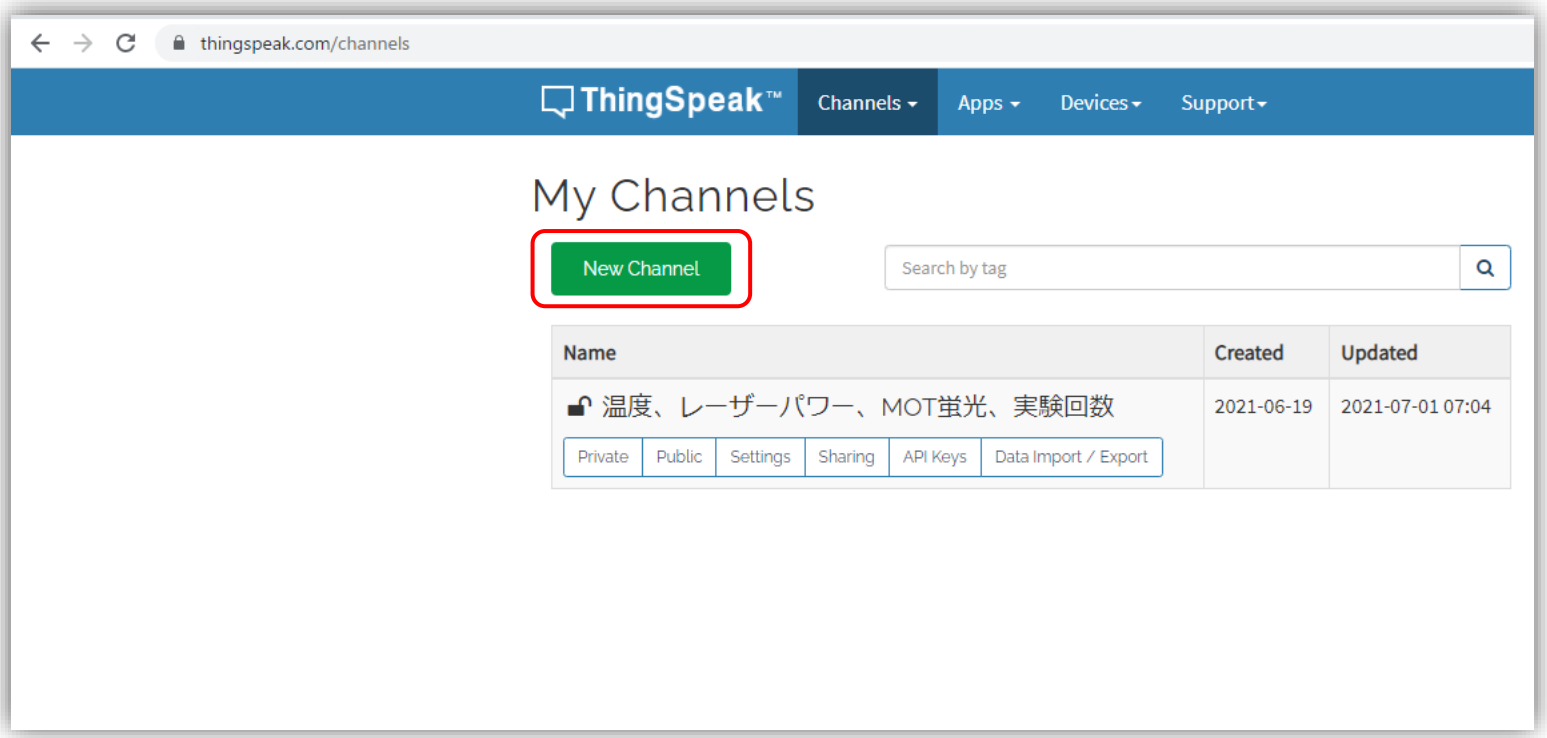
Total: ¥ 36,500/year

Purchase

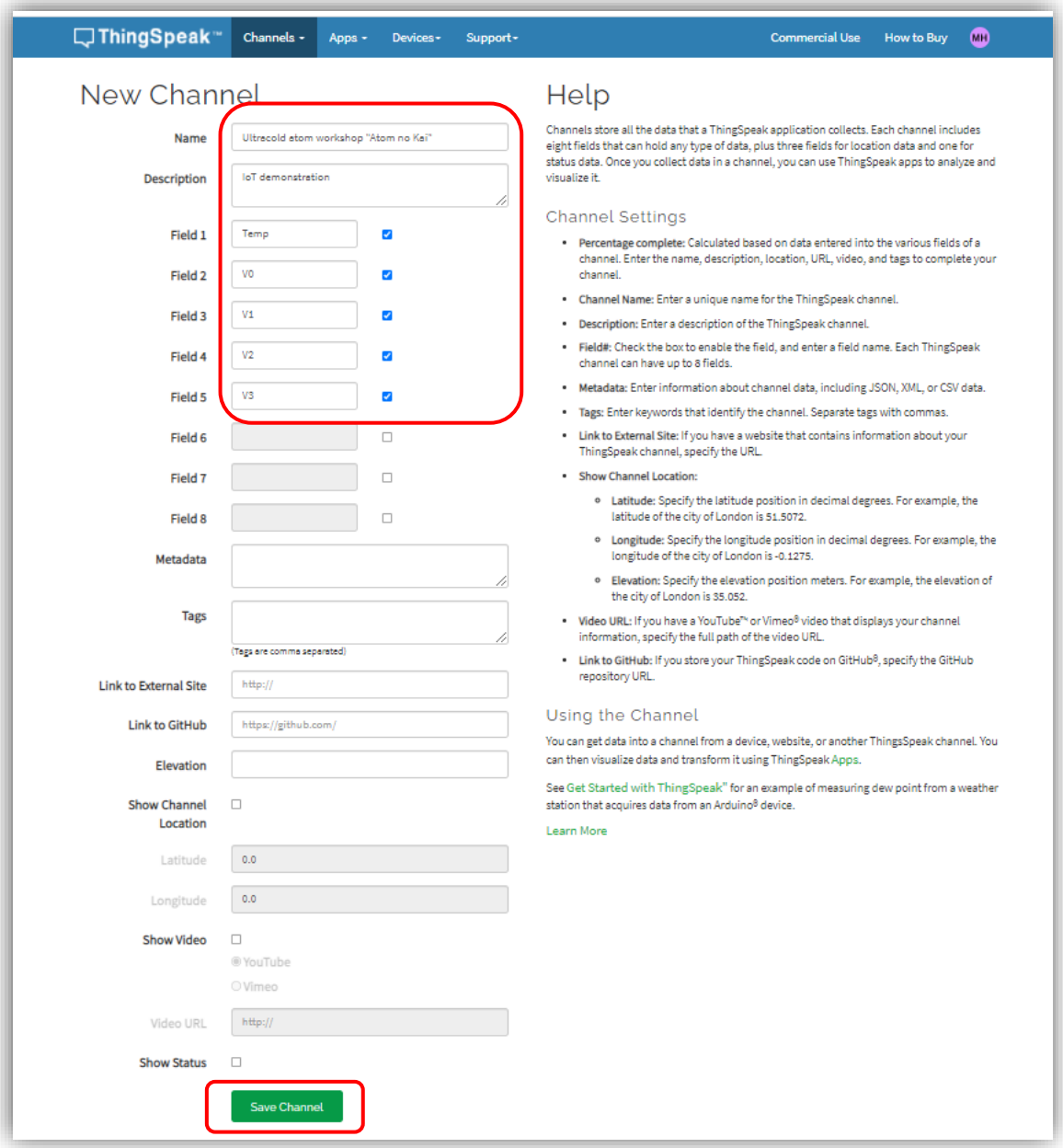
You will be taken to the MathWorks store to complete your purchase.

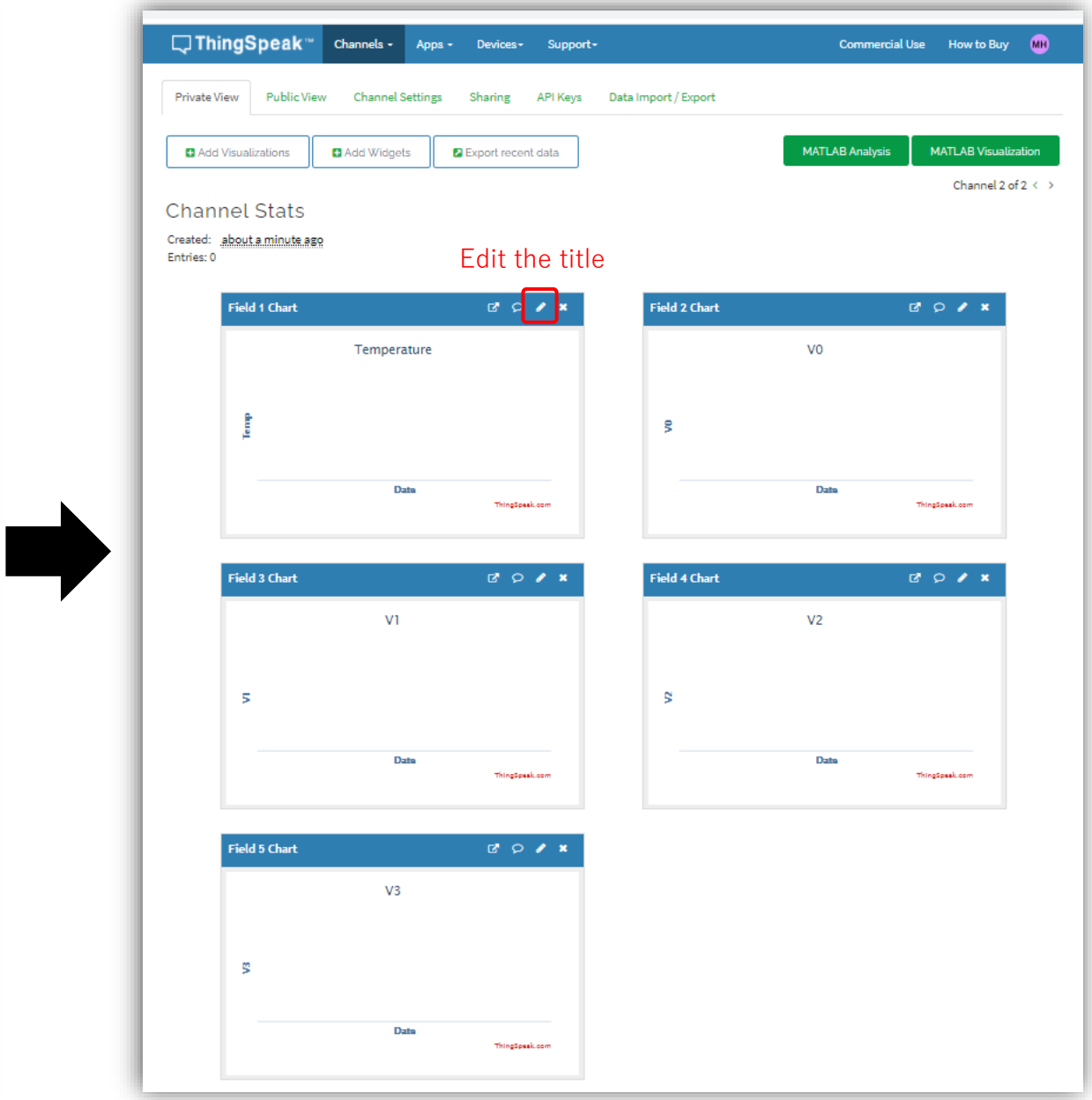
Upload data to the IOT cloud server





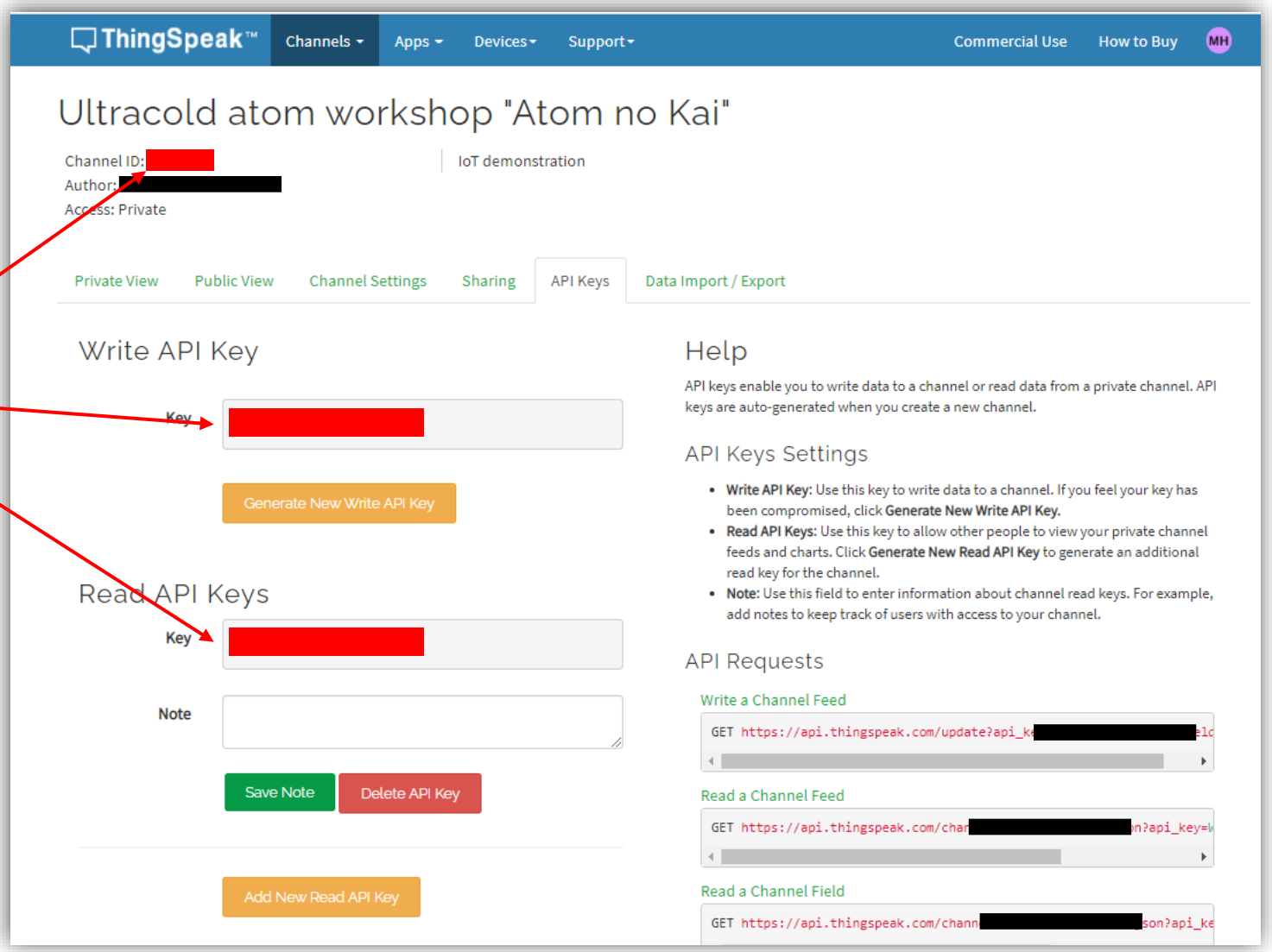
Upload data to the IOT cloud server





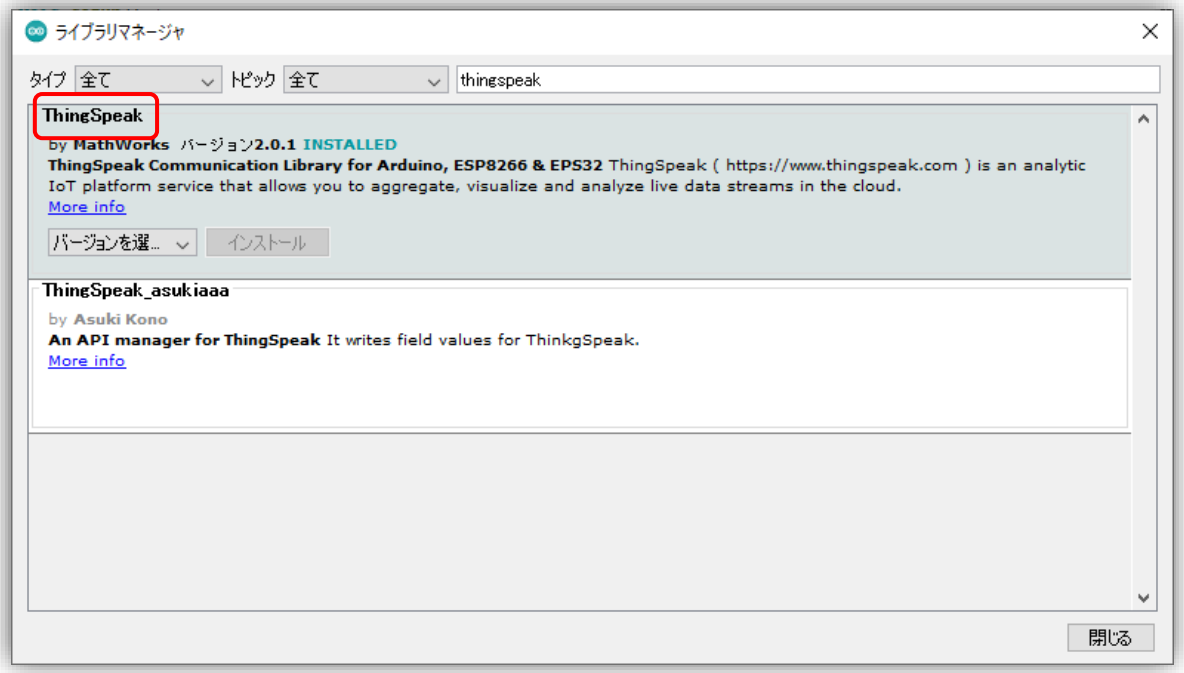
Upload data to the IOT cloud server

These three information are needed to read and write data

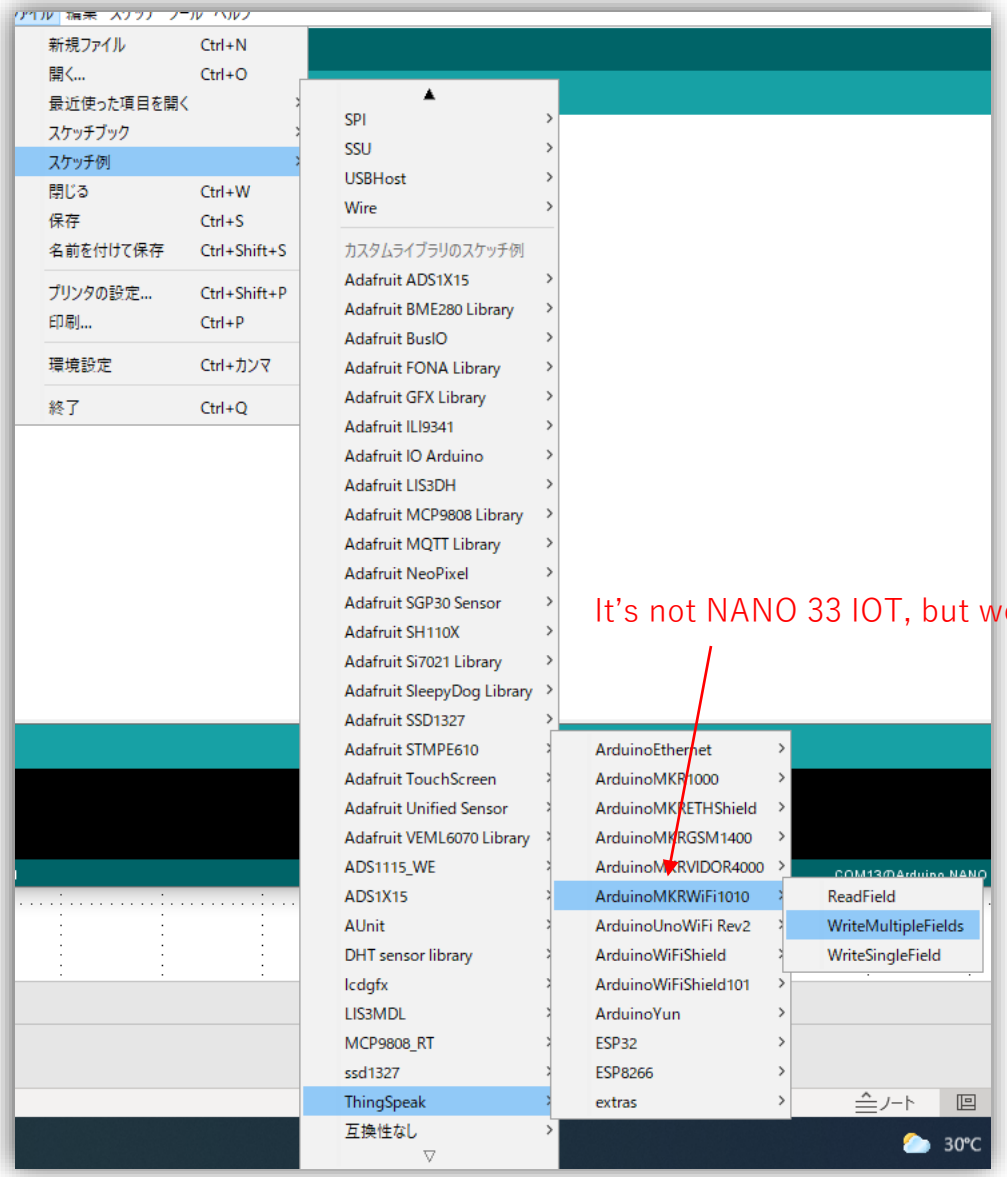


Upload data to the IOT cloud server

- Install the library for ‘ThingSpeak’



- Open the sample code, cut, paste, and edit



Upload data to the IOT cloud server

- Open the sample code, cut, paste, and edit

Hello_Temperature_ADC_Wifi

```
#include <Adafruit_SSD1327.h>
#include <Wire.h>
#include "Adafruit_MCP9808.h"
#include <ADS1115_WE.h>
#include <SPI.h>
#include <WiFiNINA.h>
#include "ThingSpeak.h" // always include thingspeak header file after other header files and custom macros

char ssid[] = "Buffalo-G-9330";        // your network SSID (name)
char pass[] = "[redacted]";    // your network password (use for WPA, or use as key for WEP)
int status = WL_IDLE_STATUS;     // the WiFi radio's status

WiFiClient  client;

unsigned long myChannelNumber = [redacted];
const char * myWriteAPIKey = "[redacted]";

// I2C
#define OLED_RESET -1
Adafruit_SSD1327 display(128, 128, &Wire, OLED_RESET, 1000000);

// Create the MCP9808 temperature sensor object
Adafruit_MCP9808 tempsensor = Adafruit_MCP9808();

// Create the ADS1115 ADC object
ADS1115_WE adc = ADS1115_WE(0x48);

// the setup routine runs once when you press reset:
void setup() {

  ThingSpeak.begin(client);  //Initialize ThingSpeak

  display.begin(0x3D);
  display.clearDisplay();
  display.setTextSize(1);
  display.display();
  display.setCursor(0, 0);
  display.setTextSize(1);
```

Your channel ID

Your Write API Key

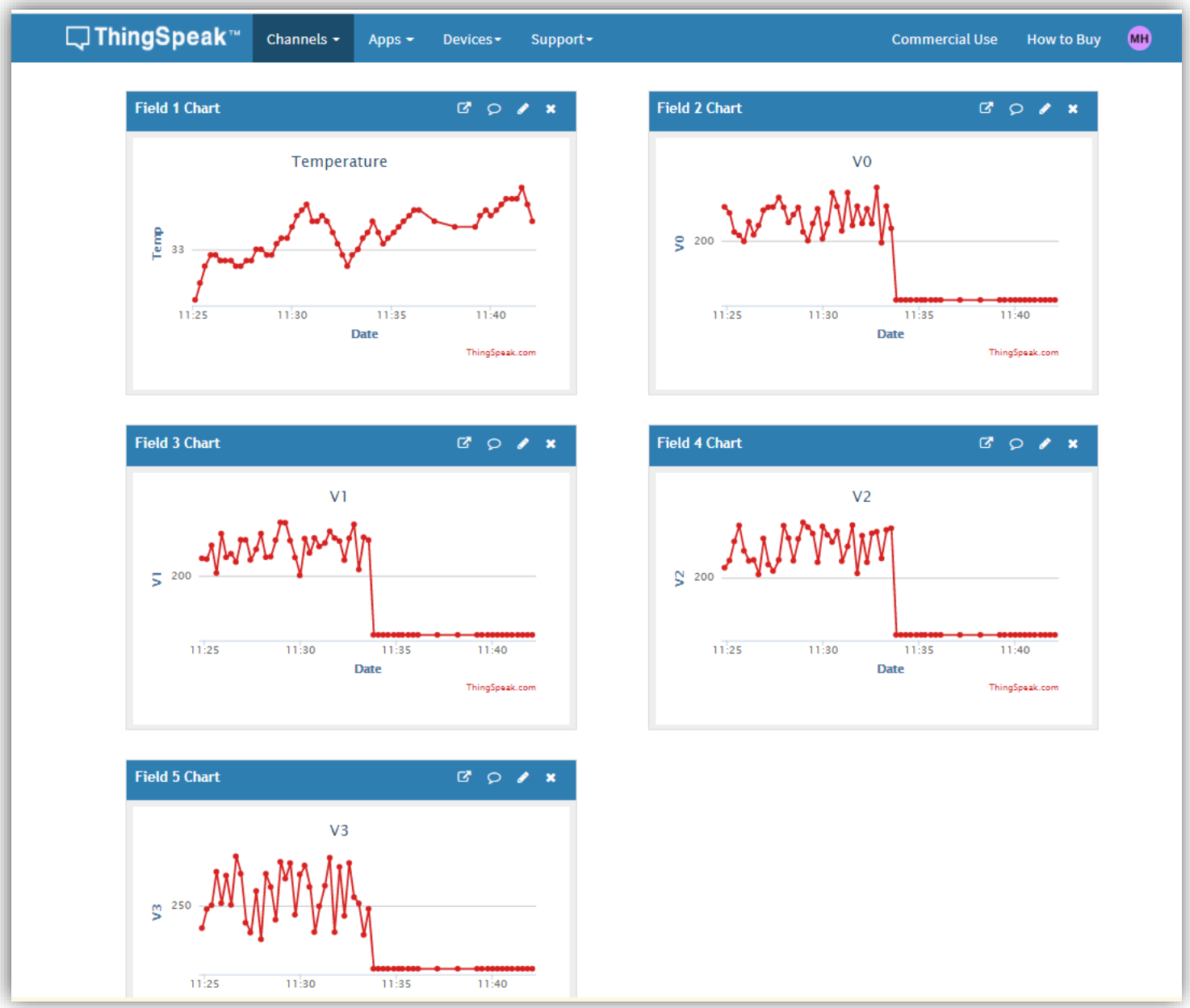
If you forget it, your Arduino freezes.

⋮

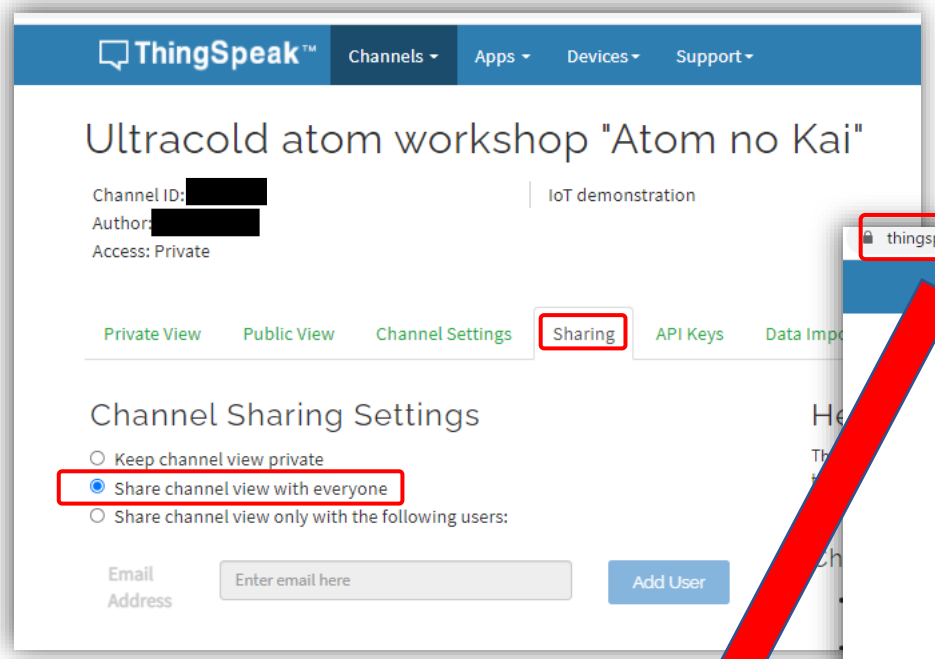
```
// the loop routine runs over and over again forever:
void loop() {

  float c = tempsensor.readTempC();
  float V0 = readChannel(ADS1115_COMP_0_GND);
  float V1 = readChannel(ADS1115_COMP_1_GND);
  float V2 = readChannel(ADS1115_COMP_2_GND);
  float V3 = readChannel(ADS1115_COMP_3_GND);

  display.clearDisplay();

  display.setCursor(0, 0);

  printWifiData();
  display.println("");

  display.print("T : "); display.print(c, 3); display.println(" [C]");
  display.println("");
  display.print("V0 : "); display.print(V0, 1); display.println(" [mV]");
  display.print("V1 : "); display.print(V1, 1); display.println(" [mV]");
  display.print("V2 : "); display.print(V2, 1); display.println(" [mV]");
  display.print("V3 : "); display.print(V3, 1); display.println(" [mV]");
  display.println("");

  ThingSpeak.setField(1, c);
  ThingSpeak.setField(2, V0);
  ThingSpeak.setField(3, V1);
  ThingSpeak.setField(4, V2);
  ThingSpeak.setField(5, V3);

  int x = ThingSpeak.writeFields(myChannelNumber, myWriteAPIKey);
  if(x == 200){
    display.println("Channel update successful.");
  }

  display.display();

  delay(500);        // delay in between reads for stability
}
```

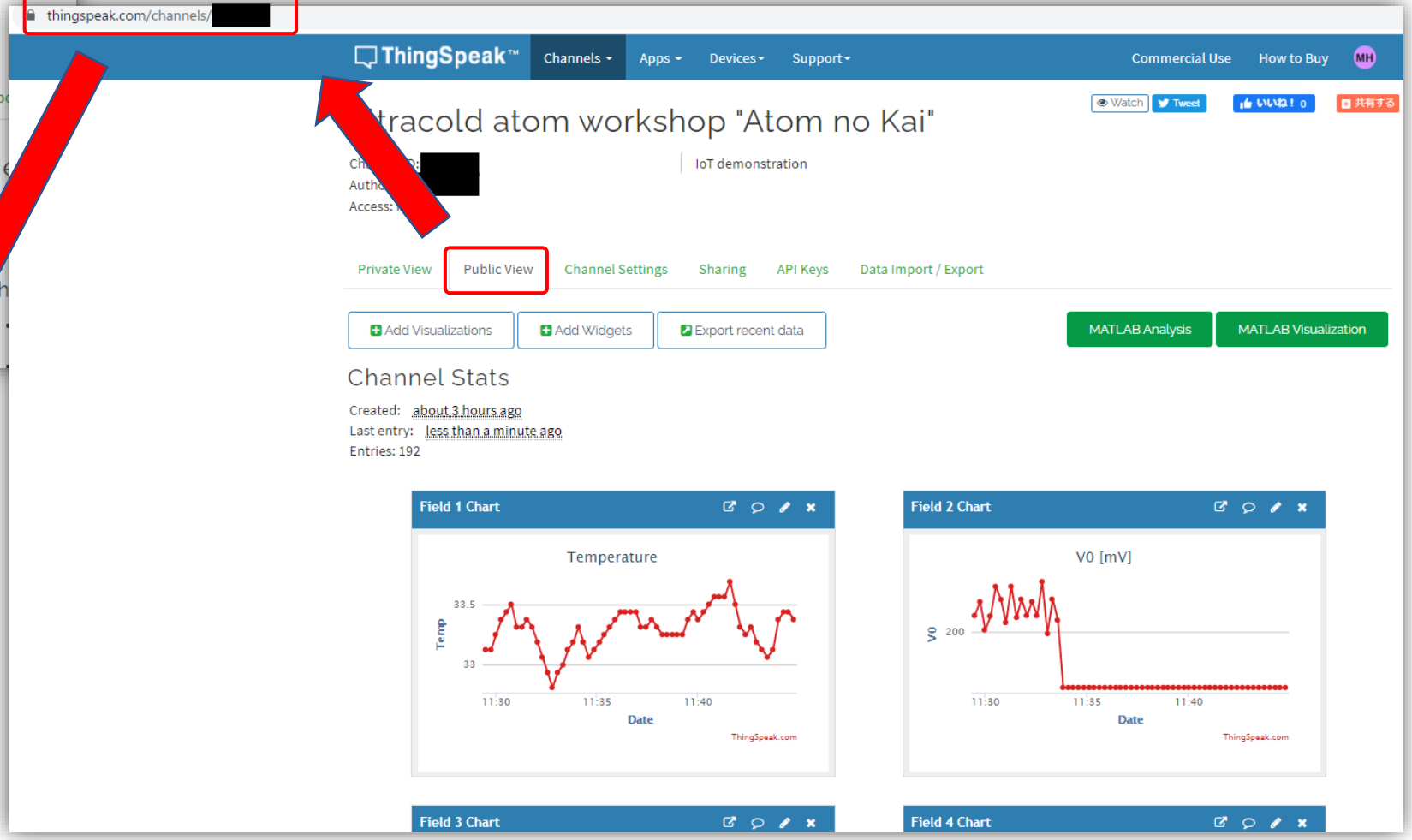
⋮

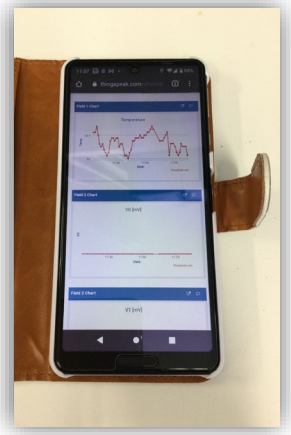
Upload data to the IOT cloud server

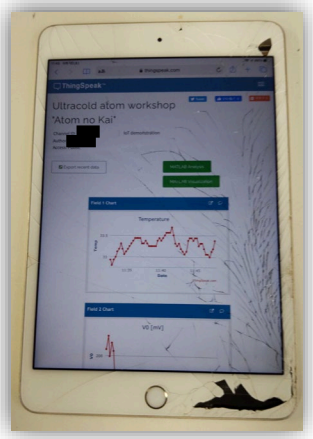


Upload data to the IOT cloud server

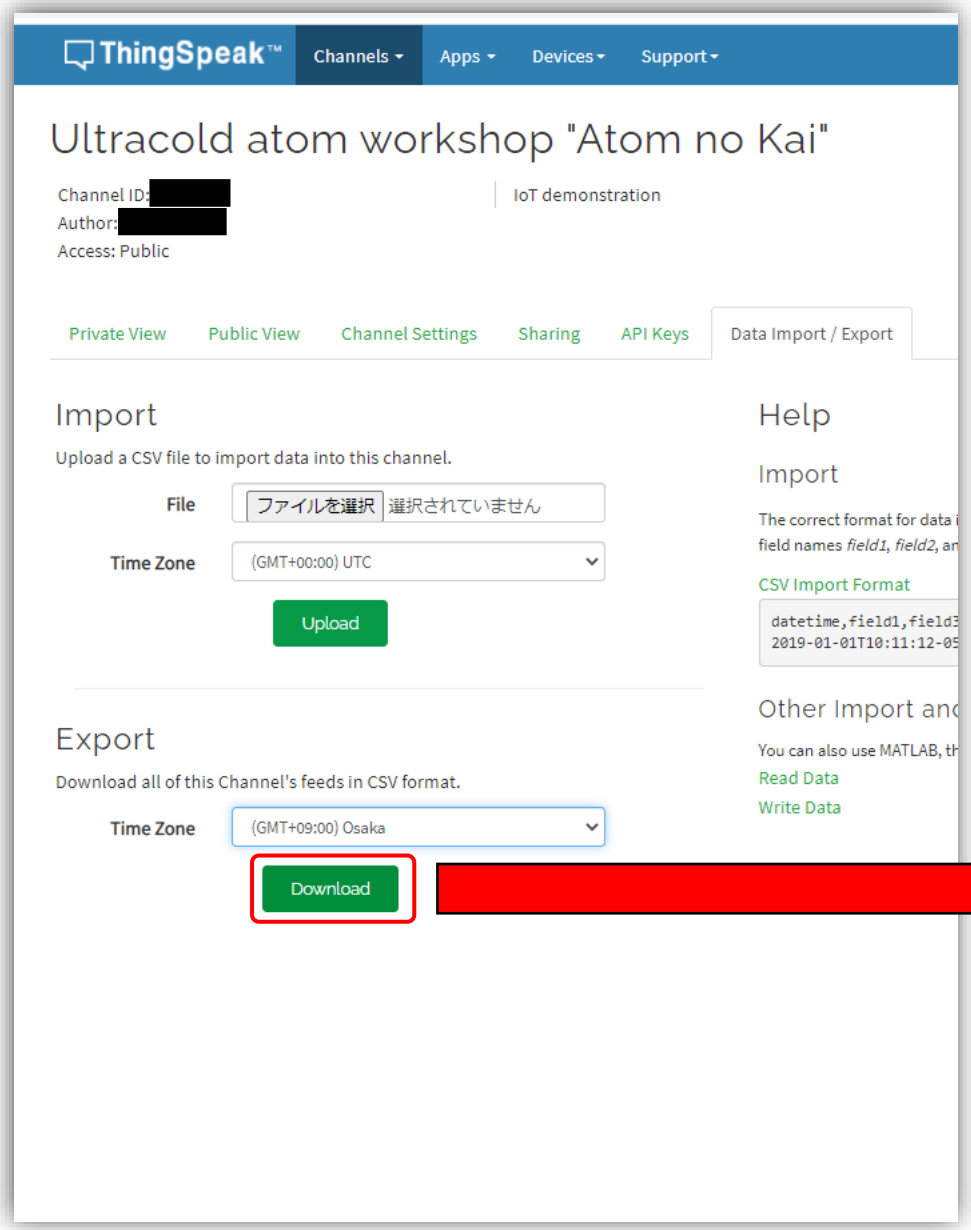


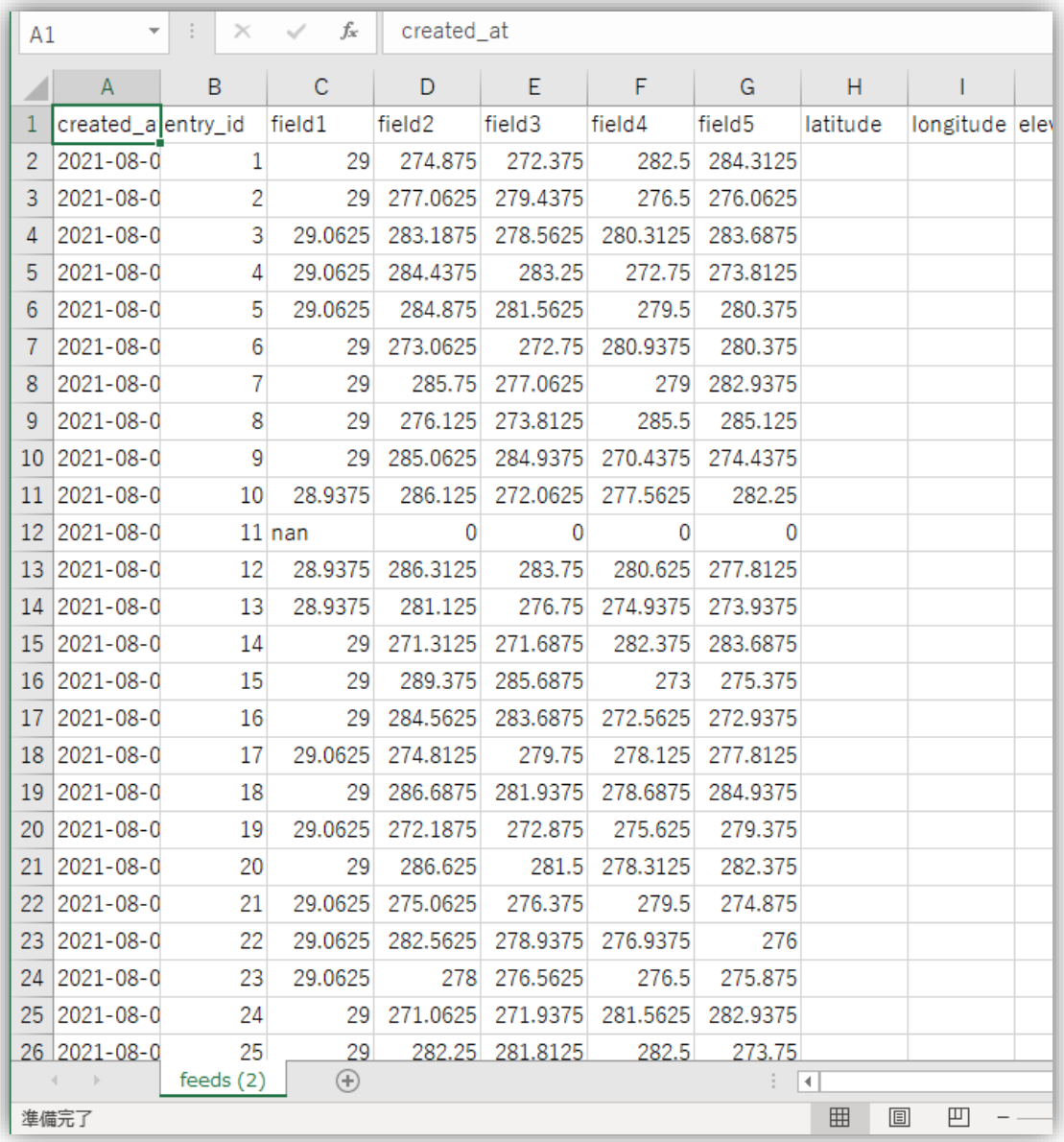






Download data from the IOT cloud server





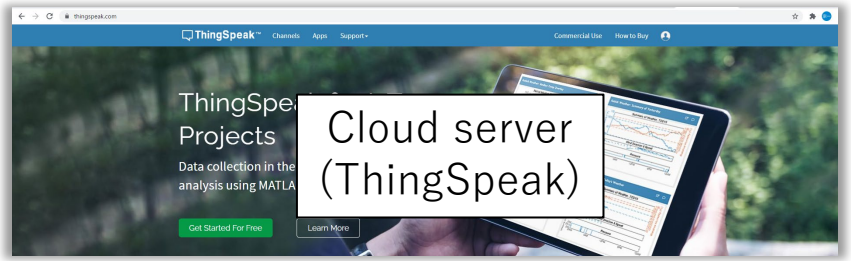
A1 | created_at

	A	B	C	D	E	F	G	H	I
1	created_a	entry_id	field1	field2	field3	field4	field5	latitude	longitude
2	2021-08-0	1	29	274.875	272.375	282.5	284.3125		
3	2021-08-0	2	29	277.0625	279.4375	276.5	276.0625		
4	2021-08-0	3	29.0625	283.1875	278.5625	280.3125	283.6875		
5	2021-08-0	4	29.0625	284.4375	283.25	272.75	273.8125		
6	2021-08-0	5	29.0625	284.875	281.5625	279.5	280.375		
7	2021-08-0	6	29	273.0625	272.75	280.9375	280.375		
8	2021-08-0	7	29	285.75	277.0625	279	282.9375		
9	2021-08-0	8	29	276.125	273.8125	285.5	285.125		
10	2021-08-0	9	29	285.0625	284.9375	270.4375	274.4375		
11	2021-08-0	10	28.9375	286.125	272.0625	277.5625	282.25		
12	2021-08-0	11	nan	0	0	0	0		
13	2021-08-0	12	28.9375	286.3125	283.75	280.625	277.8125		
14	2021-08-0	13	28.9375	281.125	276.75	274.9375	273.9375		
15	2021-08-0	14	29	271.3125	271.6875	282.375	283.6875		
16	2021-08-0	15	29	289.375	285.6875	273	275.375		
17	2021-08-0	16	29	284.5625	283.6875	272.5625	272.9375		
18	2021-08-0	17	29.0625	274.8125	279.75	278.125	277.8125		
19	2021-08-0	18	29	286.6875	281.9375	278.6875	284.9375		
20	2021-08-0	19	29.0625	272.1875	272.875	275.625	279.375		
21	2021-08-0	20	29	286.625	281.5	278.3125	282.375		
22	2021-08-0	21	29.0625	275.0625	276.375	279.5	274.875		
23	2021-08-0	22	29.0625	282.5625	278.9375	276.9375	276		
24	2021-08-0	23	29.0625	278	276.5625	276.5	275.875		
25	2021-08-0	24	29	271.0625	271.9375	281.5625	282.9375		
26	2021-08-0	25	29	282.25	281.8125	282.5	273.75		

feeds (2)

準備完了

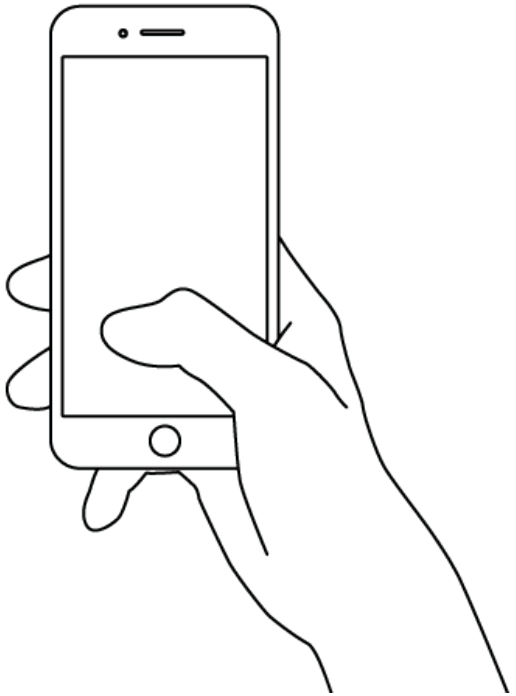
Today's goal



Cloud server (ThingSpeak)

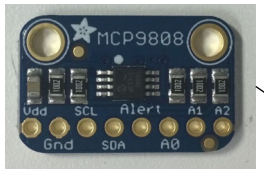
Internet

Outside your laboratory



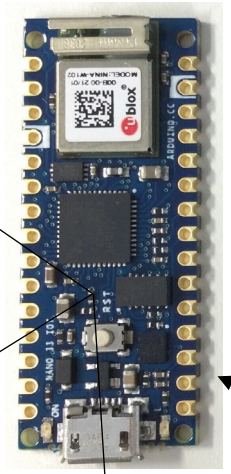
Internet

Inside your laboratory



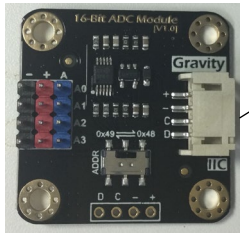
Temperature measurement

I2C



Wi-Fi

Wi-Fi router



I2C

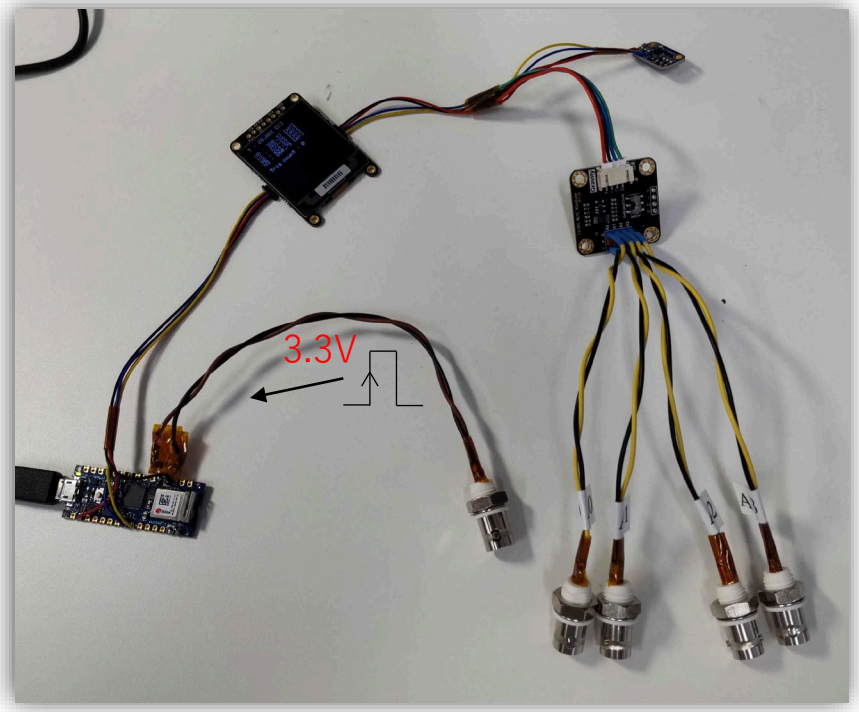
Voltage measurement
(laser power, MOT fluorescence)

I2C

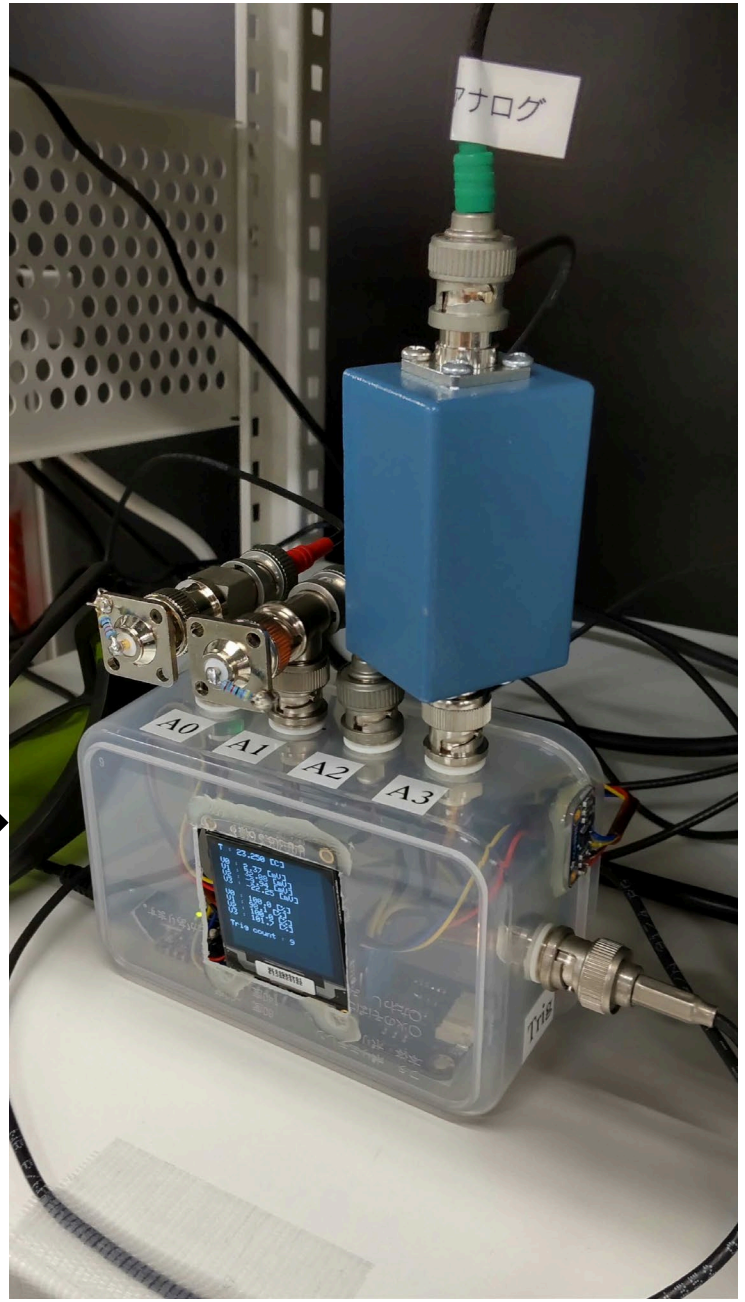
Displaying experimental conditions on an OLED

External trigger for upload data to the cloud

Upload by external triggers

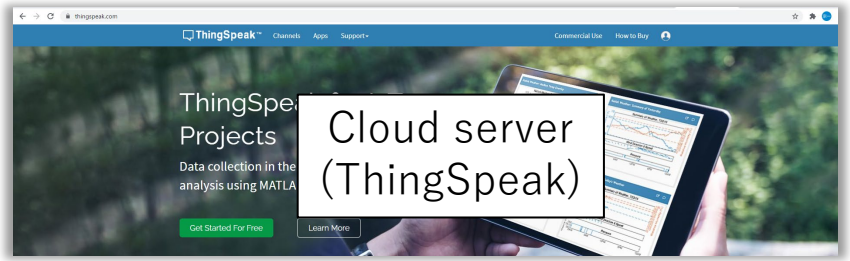


Packing



Conclusion

- Arduino is fun and practical
- LED blink, Hello world
- I2C communication to digital devices
- Wi-Fi connection
- IoT cloud server

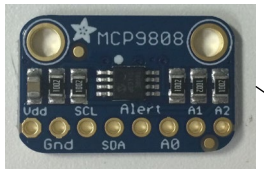


Internet

Internet

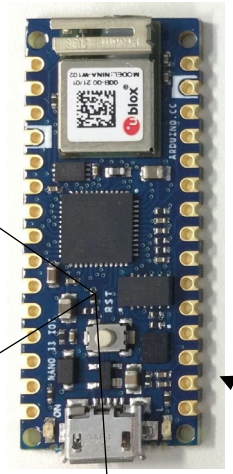
Inside your laboratory

Outside your laboratory



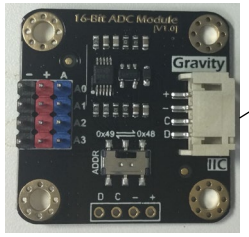
Temperature measurement

I2C



Wi-Fi

Wi-Fi router



I2C

Voltage measurement
(laser power, MOT fluorescence)

I2C

External trigger for upload
data to the cloud

Displaying experimental
conditions on an OLED