

DSI5168

A Smart Bed Exit Alarm

- Tutorial Resources:

https://github.com/Lys-0929/DSI5168_MQTT-SSL

Or you can scan the QR code



- Designed by David Lee
- Directive Agency: Industrial Development Bureau,
Ministry of Economic Affairs
- Executive Institute : Digital Service Innovation
Institute, Institute for Information Industry

Content

Chapter1

Abstract



Chapter2

Hardware Overview



Chapter3

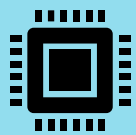
Software Overview



Chapter4

Demonstration





➤ Chapter 1 Abstract

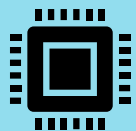
Design Purpose:

As the proportion of elders continues to rise in Taiwan, the dependency ratio has also been rising. In the case of insufficient support manpower, the demand for remote monitoring of elderly people has gradually emerged. To monitor the situation on the bed of elders is the purpose of this design. So that, even if the caregivers are not on-site, they can get the latest information of bedridden elderly people.

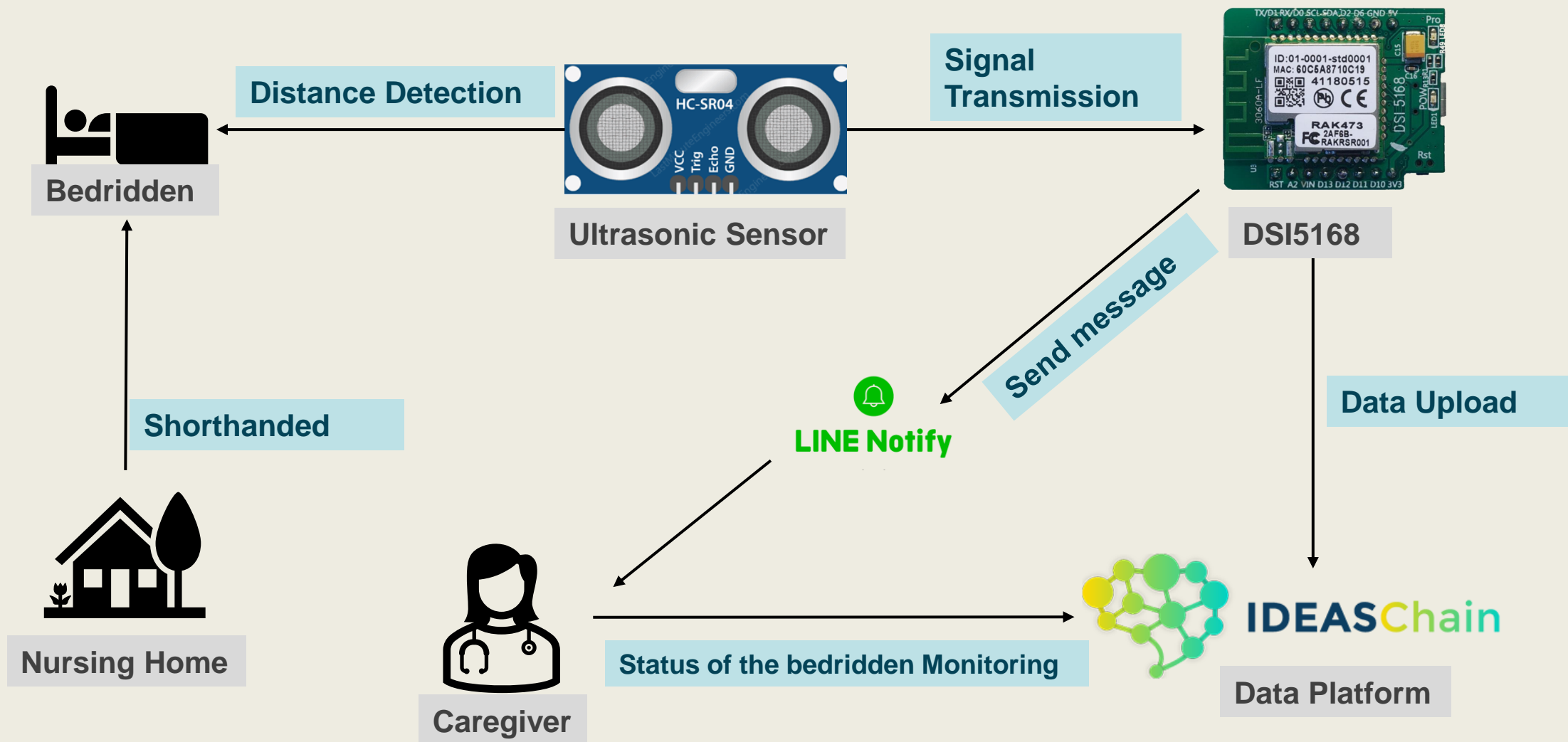
Design Methods:

The application installs the ultrasound sensor on the head of the bed. Measure the change of the distance from the patient to the head of the bed. The distance decreasing as the patient gets up. If the sensor detects the distance decreasing, LINE will publish an alarm message to the caregiver.

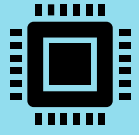




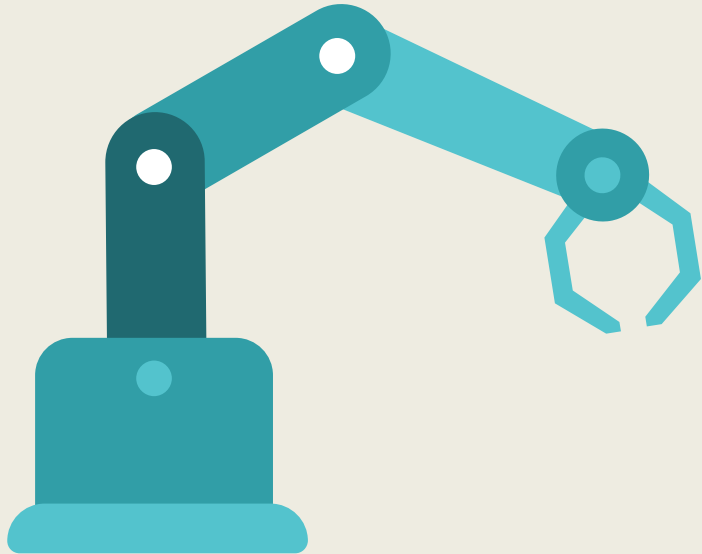
> 1-1 Schematic diagram of the Project



(Image Source : IDEAS Chain)



➤ Chapter 2 Hardware Overview



2-1

DSI-5168 Evaluation Board

2-2

DSI5168 Pinout, Features

2-3

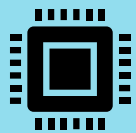
RTL8711AM Chip

2-4

Sensor Overview

2-5

Wiring HC-SR04 with DSI5168



➤ 2-1 DSI5168 Evaluation Board



Micro Control Unit

The module offers a flexible interface includes SPI, I2C, UART, PWM and ADC ports.



Wi-Fi connect

Support For LAN 802.11b/g/n
Operating frequency 2.4GHz

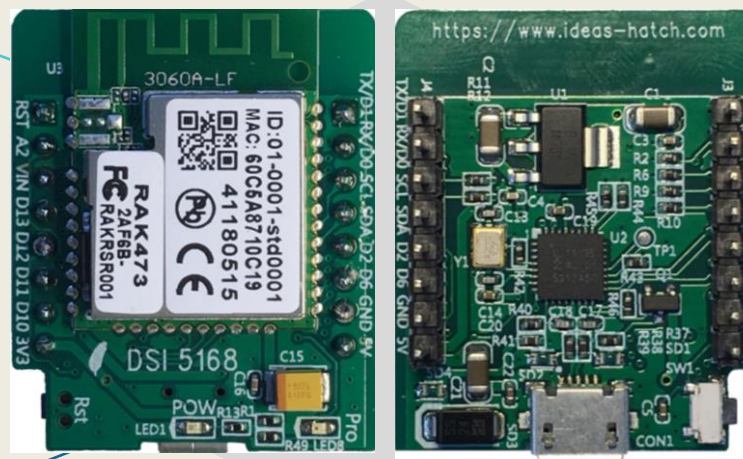
Development Enviroment

Can be compiled and burned through Arduino IDE, also compatible the development environment of Arduino

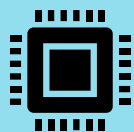


IOT

Support For MQTT and http network protocol, can be used to monitor the updated data.



Institute for Information Industry takes Ameba-RTL8711AM as the core of DSI5168. The Evaluation Board made in Taiwan is designed for IoT. It is compatible with Arduino IDE development environment that makes it more convenient to use.

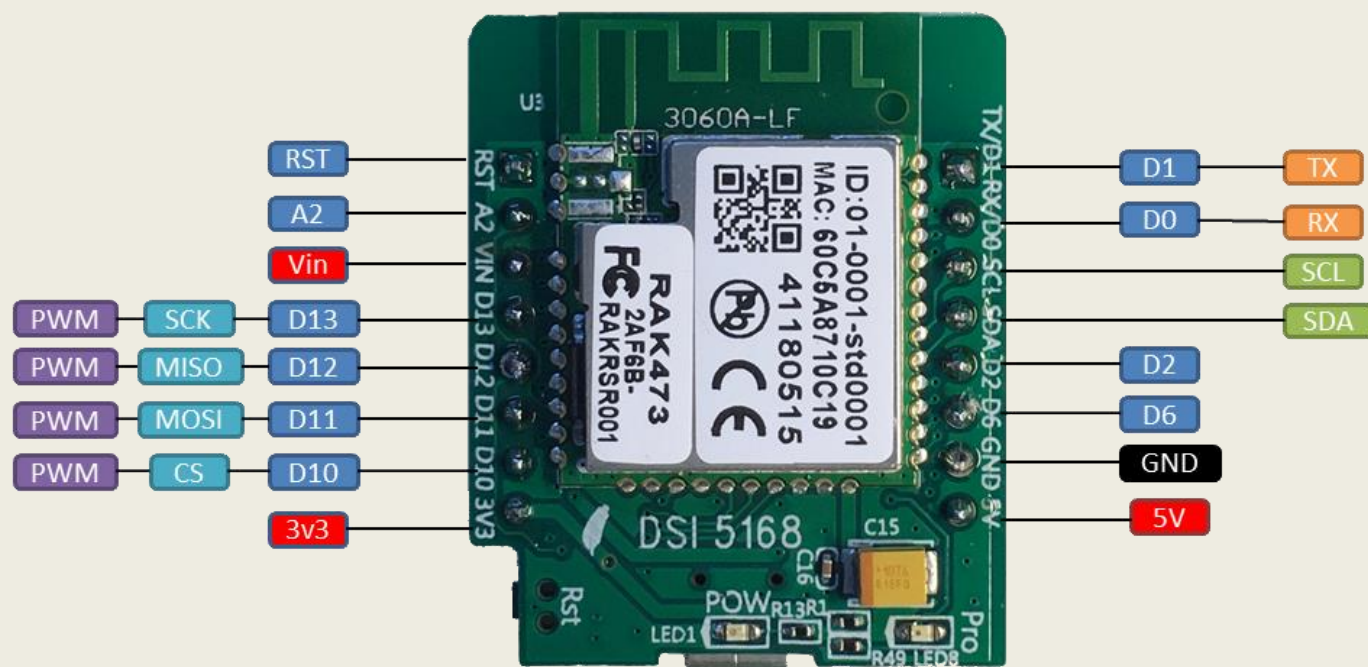


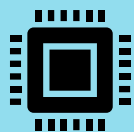
➤ 2-2 Pinout & Features of DSI5168

Board Features

硬體功能	規格
Chipset	RTL8711AM
MCU	ARM M3/166MHz
I/O	12
ROM	1MB
SRAM	512KB
Internal Flash	N/A
External Flash	2MB
ADC	1
SPI	1
UART	1
I2C	1
I2S	N/A
PWM	4
SSL	Support

- UART function
- I2C definition
- Arduino definition
- SPI definition
- PWM function





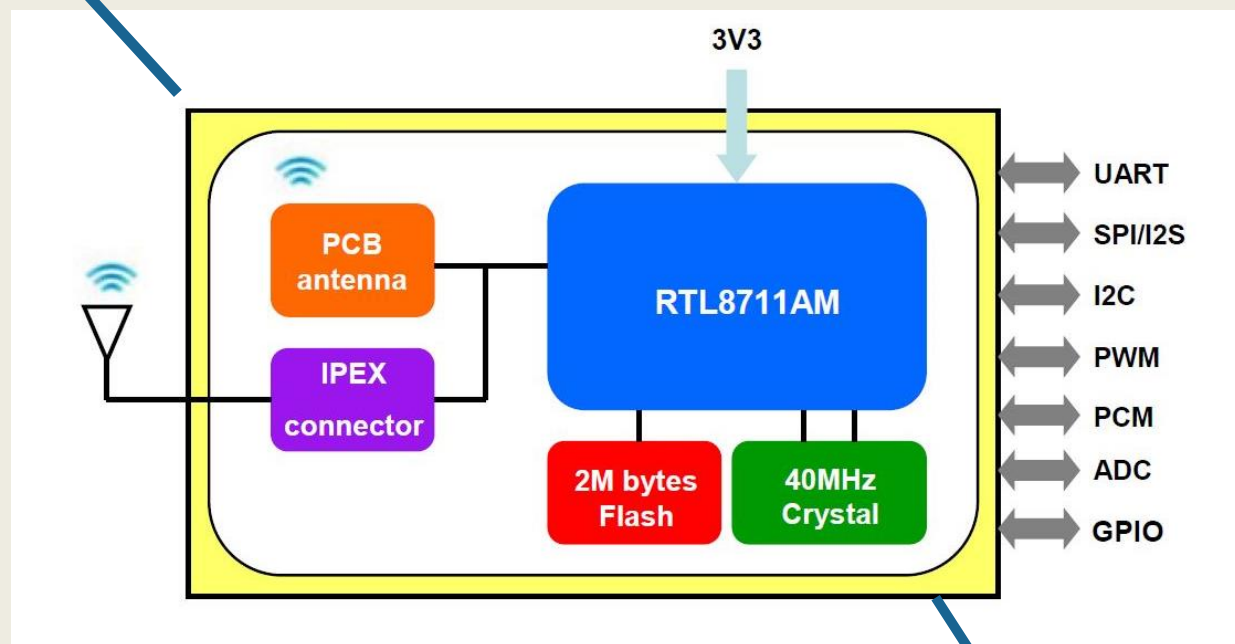
2-3



REALTEK Ameba RTL8711AM

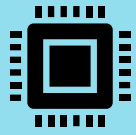
IC Data Sheet

RTL8711AM WiFi Module is a small form factor, single stream, 801.11 b/g/n WiFi module with embedded low power application processor.



General Features

- Connector ports including GPIO ports, UART, I2C, SPI, I2S, PCM, PWM, ADC.
- Integrated 2MB flash
- Small footprint: 19x22.25x2.3 mm
- IEEE 802.11b/g/n compliant



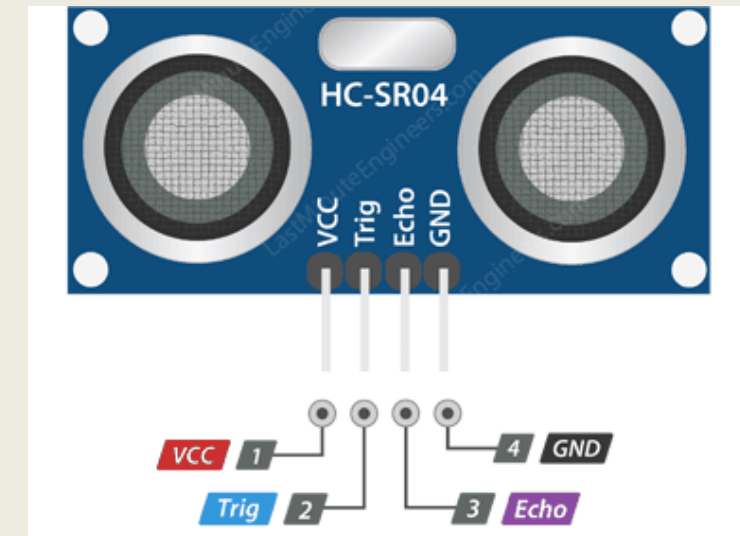
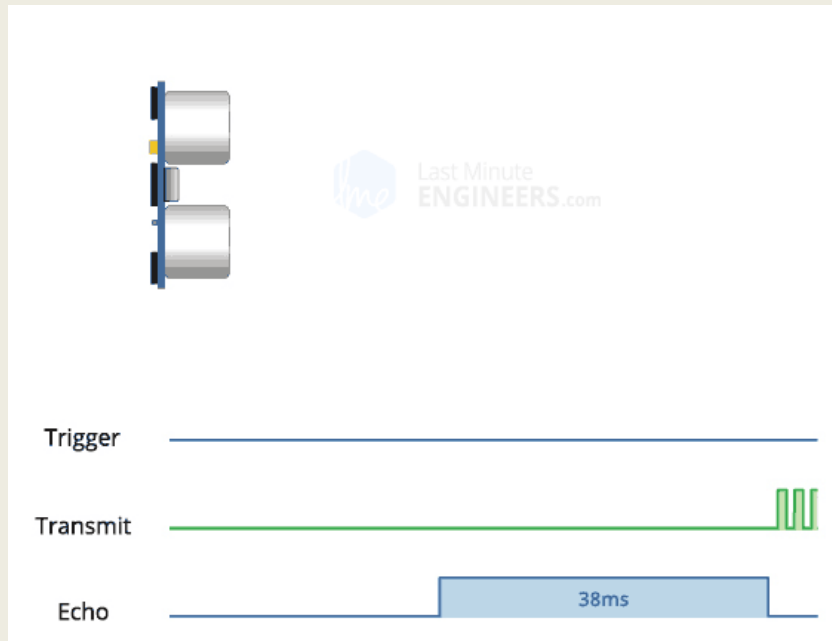
➤ 2-4 Sensor Overview

“HC-SR04” Ultrasonic sensor

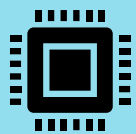
The HC-SR04 Ultrasonic distance sensor consists of two ultrasonic transducers. The one acts as a transmitter which converts electrical signal into 40 KHz ultrasonic sound pulses. The receiver listens for the transmitted pulses. If it receives them it produces an output pulse whose width can be used to determine the distance the pulse travelled.

Specifications

Operating Voltage	DC 5V
Max Range	4m
Min Range	2cm
Ranging Accuracy	3mm
Trigger Input Signal	10μS TTL pulse
Dimension	45 x 20 x 15mm



(Image Source : lastminuteengineers.com)



➤ 2-5 Wiring HC-SR04 with DSI5168

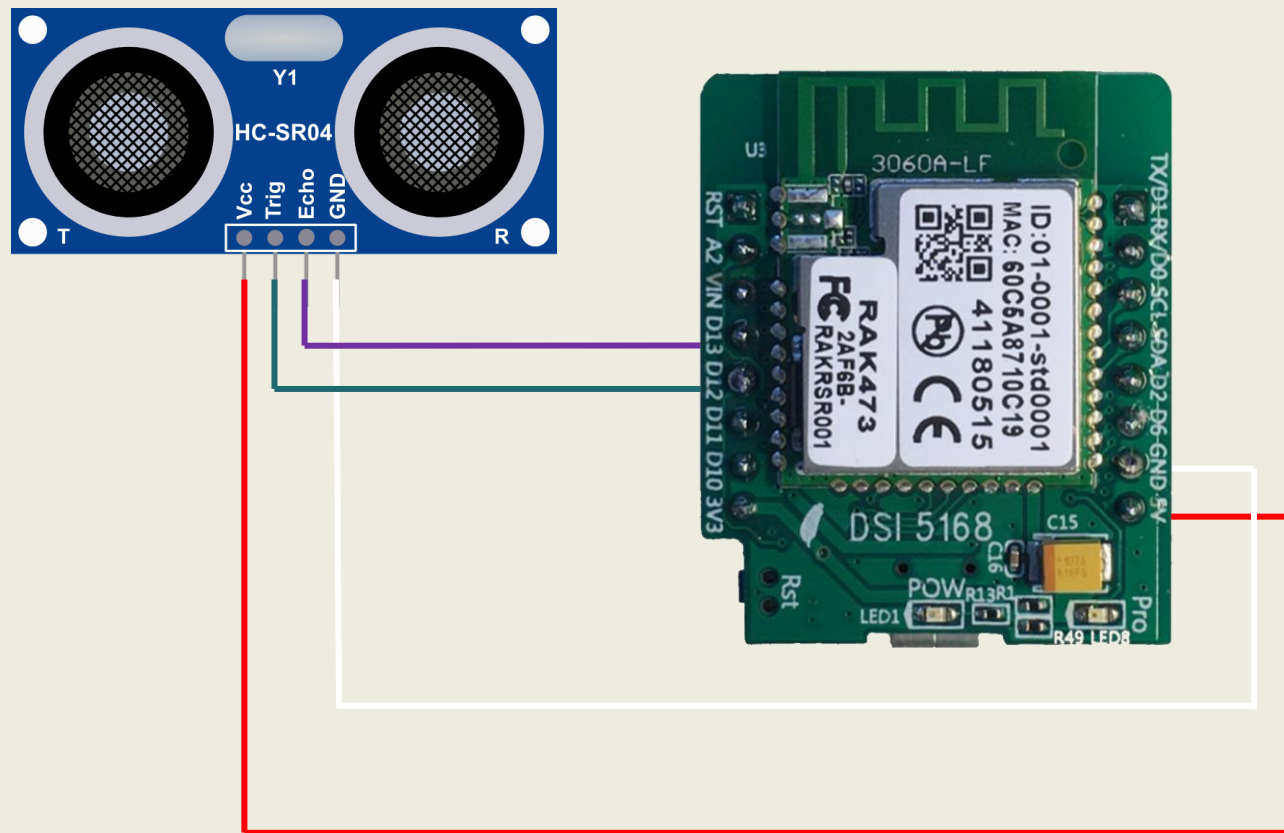
Components

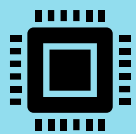
- 1.DSI5168 Evaluation Board
- 2.Ultrasonic sensor HC-SR04
- 3.Jumper wires

Pin connection

- Connect DSI5168 to power with micro usb.
- Wiring:

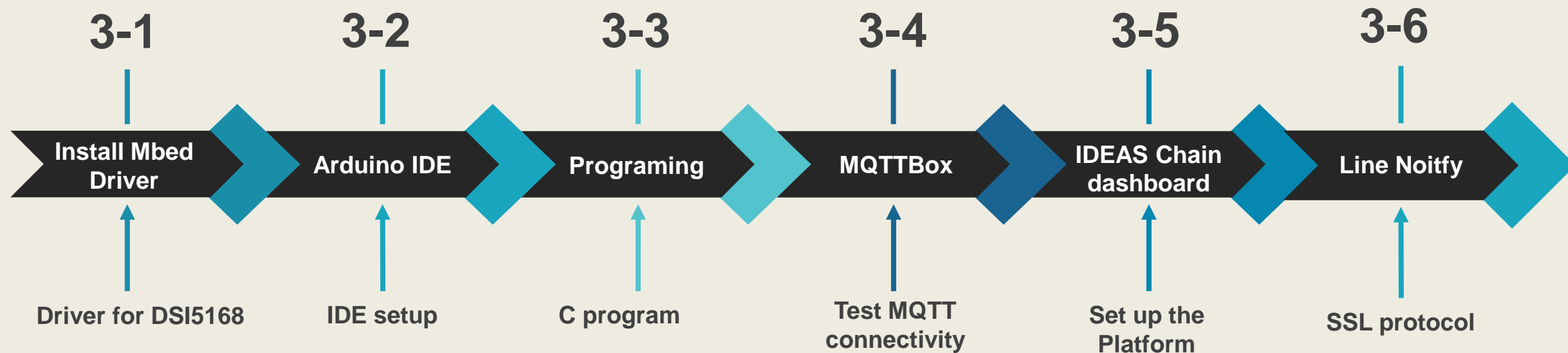
Vcc → 5V
trig → D12
echo → D13
GND → GND

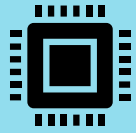




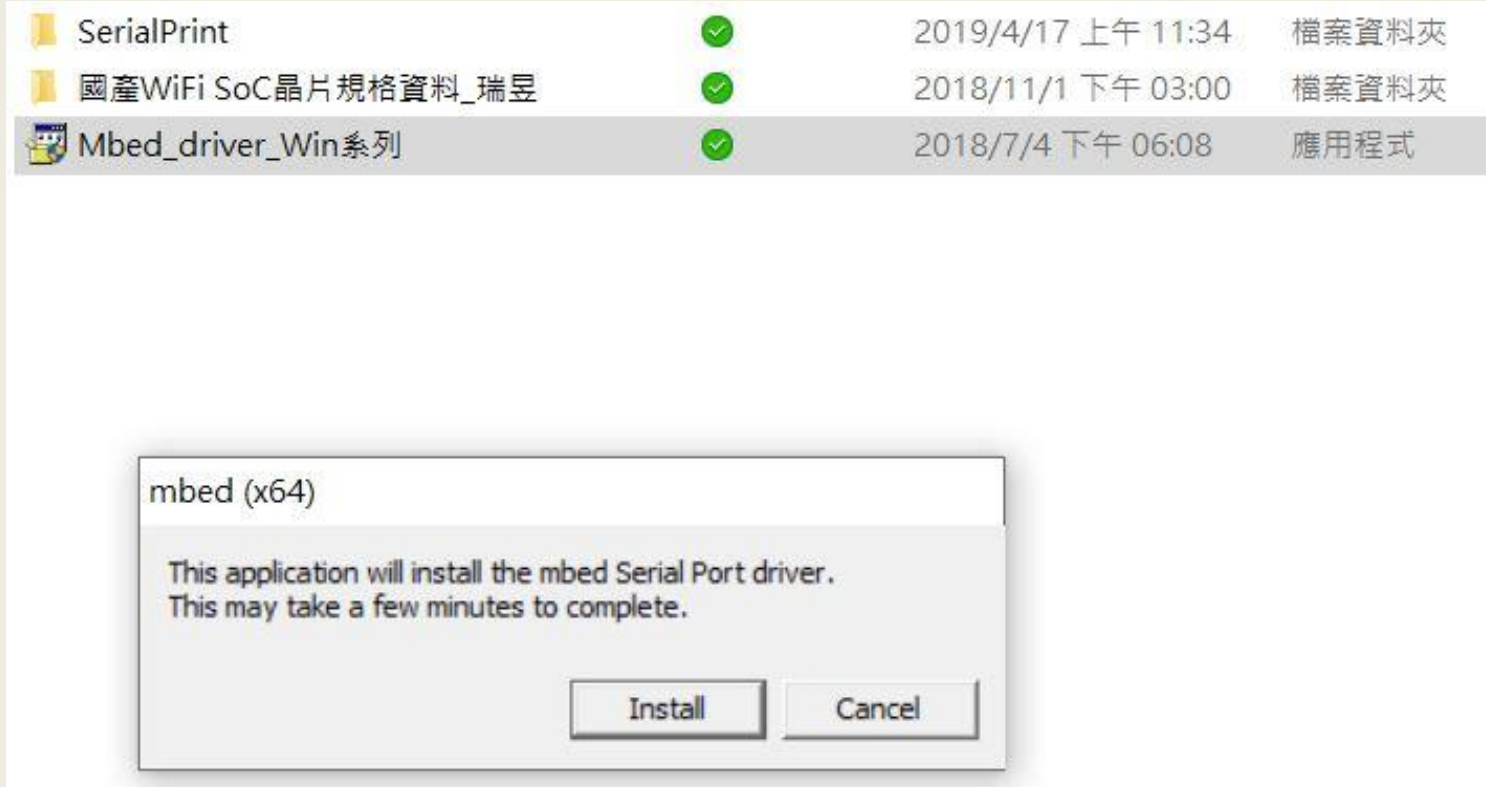
> Chapter 3 Software Overview

For Windows 10



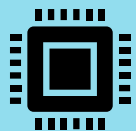


➤ 3-1 Install Mbed Driver



First, Wiring DSI5168 with your computer.


Execute the driver “mbedWinSerial_16466.exe”. Then, Open “Device Manager” to check Mbed Disk and Com Port. You’ll find the port of DSI5168.



➤ 3-2-a Install the Arduino IDE

[HARDWARE](#) [SOFTWARE](#) [CLOUD](#) [DOCUMENTATION](#) [COMMUNITY](#) [BLOG](#) [ABOUT](#)

Downloads



Arduino IDE 1.8.13

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.


Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.

DOWNLOAD OPTIONS

Windows Win 7 and newer
Windows ZIP file

Windows app Win 8.1 or 10 

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits

Mac OS X 10.10 or newer

[Release Notes](#) [Checksums \(sha512\)](#)

[HARDWARE](#) [SOFTWARE](#) [CLOUD](#) [DOCUMENTATION](#) [COMMUNITY](#) [BLOG](#) [ABOUT](#)

Support the Arduino IDE

Since the release 1.x release in March 2015, the Arduino IDE has been downloaded **50,614,238** times — impressive! Help its development with a donation.

\$3

\$5

\$10


\$25

\$50

Other

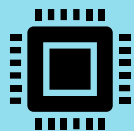
JUST DOWNLOAD

CONTRIBUTE & DOWNLOAD

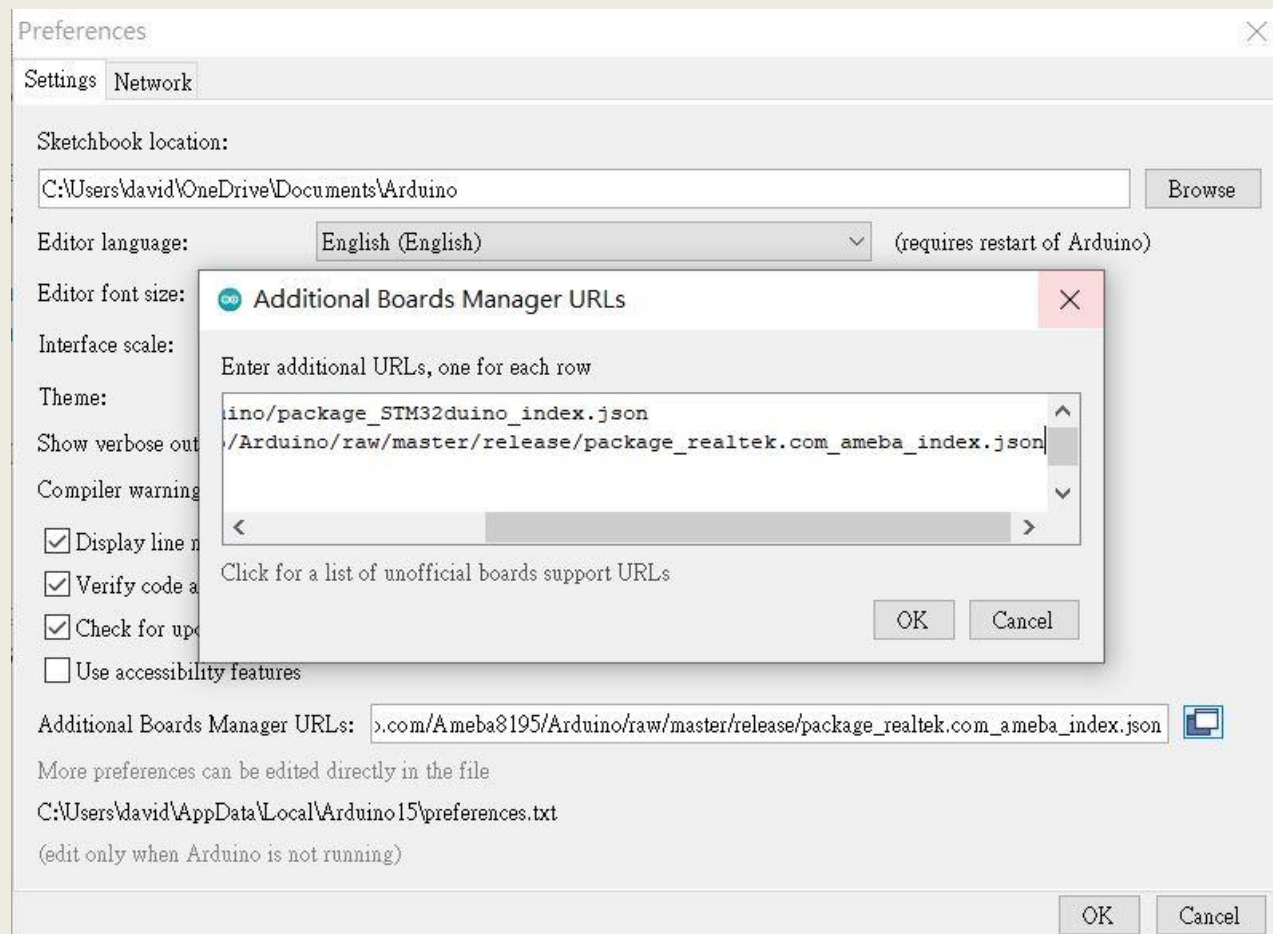
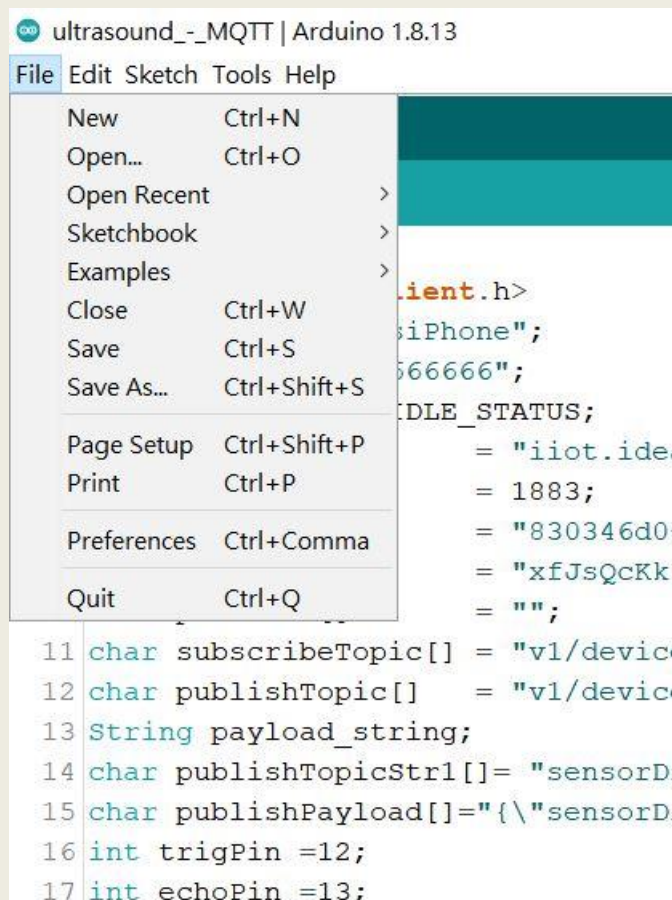


Click the following link to download Arduino IDE:
<https://www.arduino.cc/en/software>

You can choice whether you want to contribute Arduino or not. If you don't want to contribute, just click "JUST DOWNLOAD".

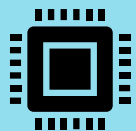


➤ 3-2-b Arduino IDE Setting

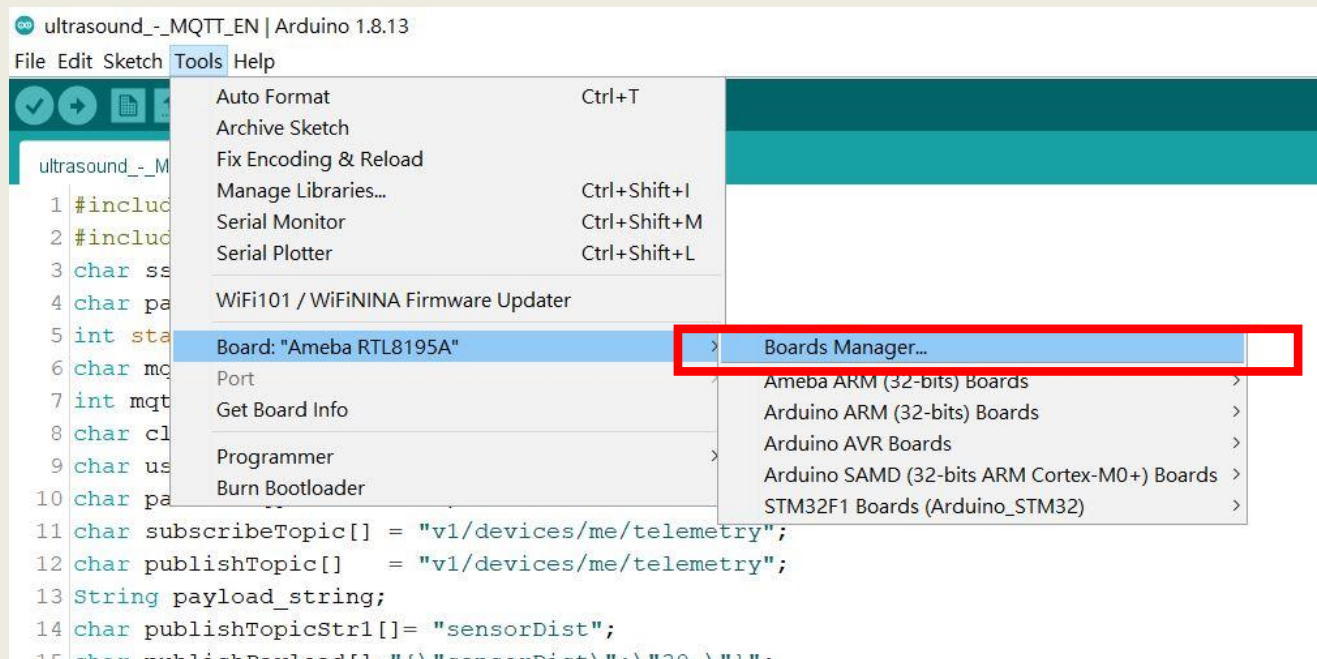


Find “Additional Boards Manager URLs “ and enter the following URL :

https://github.com/Ameba8195/Arduino/raw/master/release/package_realtek.com_ameba_index.json

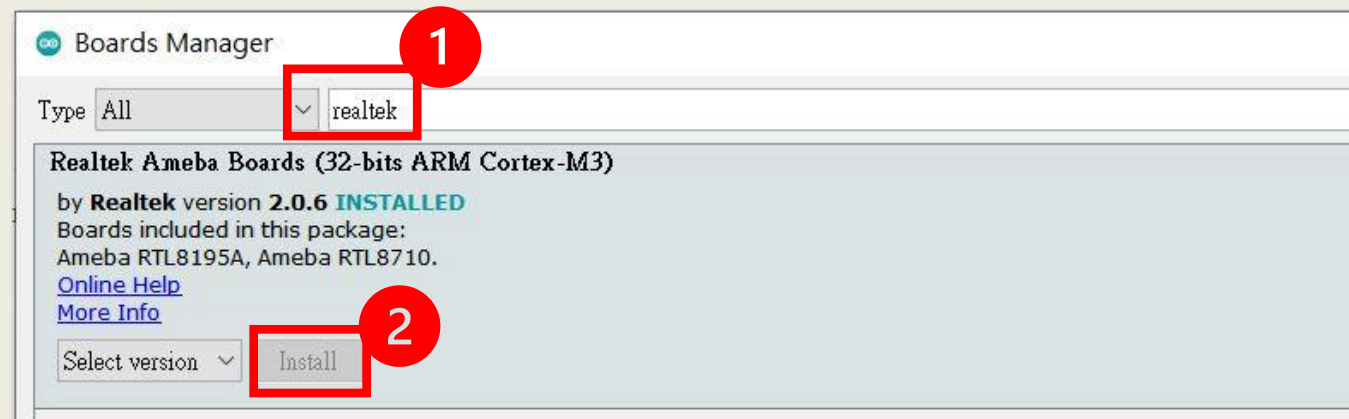


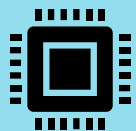
➤ 3-2-c Arduino IDE Setting



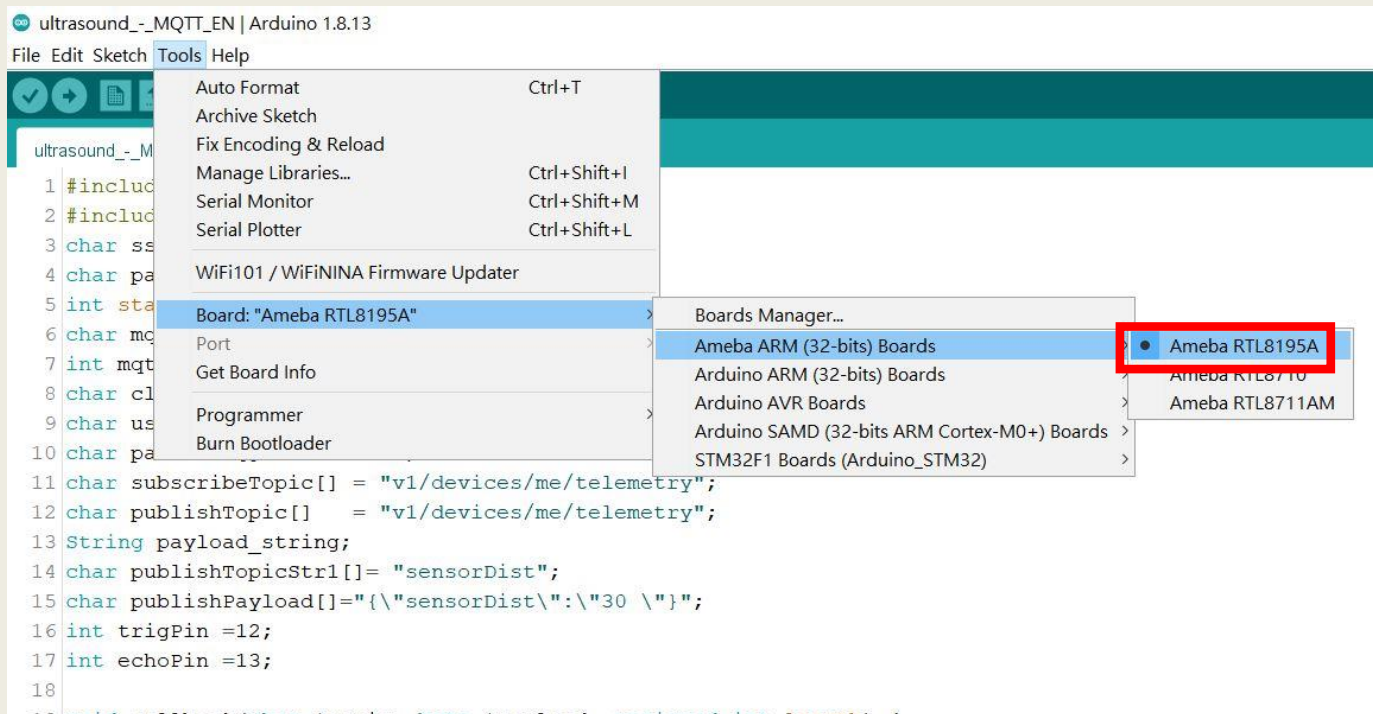
Find “Tools” and choose Boards Manager.

Enter “Realtek” and install “Realtek Ameba Boards (32-bit ARM Cortex-M3)”



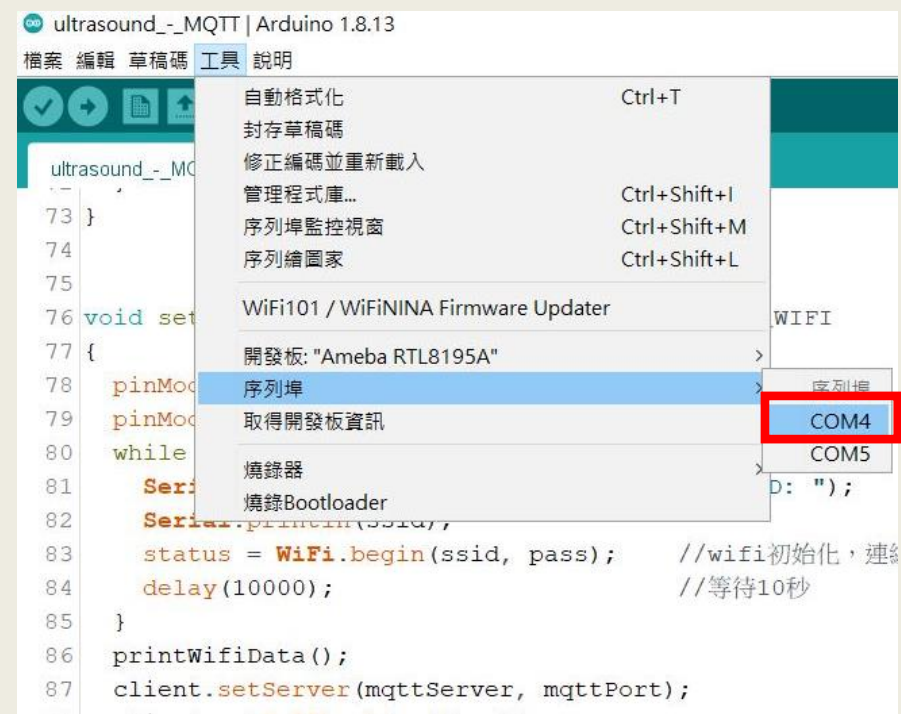


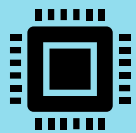
➤ 3-2-d Arduino IDE Setting



Choose Board “Ameba RTL8195A”

Choose Port “COM X”
(You can check the port from Device Manager)





> 3-3 Programing

Using Arduino IDE



Ameba Library



<WiFi.h>

This library allows the board to connect to the internet. It can serve as either a server accepting incoming connections or a client making outgoing ones.



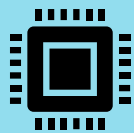
<PubSubClient.h>

This library allows you to send and receive MQTT messages. MQTT is a lightweight messaging protocol ideal for small devices.



<AmebaServo.h>

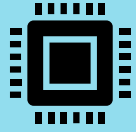
This library is the same as <servo.h>. Allows DSI5168 to control a variety of servo motors.



> 3-3-a Coding

程式碼下載: https://github.com/Lys-0929/DSI5168_MQTT-SSL

```
#include <WiFi.h>           //WIFI library
#include <PubSubClient.h>    //MQTT library
char ssid[] = "*****";     // SSID:router name
char pass[] = "*****";     // pass:router password
String Linetoken = "*****"; //Change your own "LINE Notify token"
int status = WL_IDLE_STATUS; // keep connecting
char mqttServer[] = "iiot.ideaschain.com.tw"; // Take ideaschain as server
int mqttPort = 1883;
char clientId[] = "*****"; // MQTT client ID. Create an unique ID.
char username[] = "*****"; // device access token(change your own access token of IDEASChain)
char password[] = ""; // don't need to set up
char subscribeTopic[] = "v1/devices/me/telemetry"; //fixed topic, do not modify
char publishTopic[] = "v1/devices/me/telemetry"; //as the same as subscribeTopic
String payload_string;
char publishTopicStr1[] = "sensorDist";
char publishPayload[] = "{\"sensorDist\":\"30\"}";
WiFiSSLClient client1;
char host[] = "notify-api.line.me"; //LINE Notify API URL
int trigPin = 12;
int echoPin = 13;
```

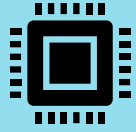



3-3-b Coding

```
void callback(char *topic, byte *payload, unsigned int length) { //recept the data from server
    Serial.print("Message arrived in topic: ");
    Serial.println(topic);
    Serial.print("Message:");
    for (int i=0; i<length; i++) {
        Serial.print( (char) payload[i]);          // convert *byte to string
    }

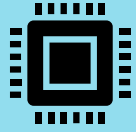
    Serial.println();
    Serial.println("-----");
}

WiFiClient wifiClient;
PubSubClient client(wifiClient);                  //define the client's name
```



3-3-c Coding

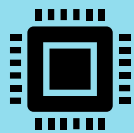
```
void publishData( char*publishTopicStr, float sensorValue){
char sensorDist [30];
sprintf (sensorDist , "{\\\"%s \\\":\\\"%.2f \\\"}", publishTopicStr,sensorValue);
Serial.println(sensorDist);
while(!client.connected()){
  Serial.println("Attempting MQTT connection Attempt to connect...");
  if(client.connect (clientId, username, password)){
    Serial.println("MQTT connected");
    client.publish(publishTopic,sensorDist);
    client.subscribe(subscribeTopic);
  }
  else{
    Serial.print("failed rc= ");
    Serial.print(client.state());
    Serial.println("try again in 5 seconds ");
    delay(5000);
  }
}
```



3-3-d Coding

```
void reconnect() {                // client connect to the MQTT server

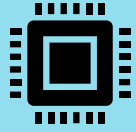
while (!client.connected()) {      //while(disconnect),then run the loop continually
  Serial.println("Attempting MQTT connection...");
  if (client.connect(clientId, username, password)) { // try to connect
    Serial.println("MQTT connected");                //after connected, publish the topic & payload
    client.publish(publishTopic, "payload_string");
    client.subscribe(subscribeTopic);                //resubscribe the topic
  } else {
    Serial.print("failed, rc=");
    Serial.print(client.state());
    Serial.println(" try again in 5 seconds");
    delay(5000);                                     //wating for 5 second to reconnect
  }
}
}
```



3-3-e Coding

```
void setup()           //set up the pinmode and WIFI
{
  pinMode(trigPin ,OUTPUT);
  pinMode(echoPin ,INPUT);
  while (status != WL_CONNECTED) {
    Serial.print("Attempting to connect to SSID: ");
    Serial.println(ssid);
    status = WiFi.begin(ssid, pass);  //initialize wifi setting
    delay(10000);                     //waiting for WiFi connecting for 10 second
  }
  printWifiData();
  client.setServer(mqttServer, mqttPort);
  client.setCallback(callback);
  delay(1500);
}

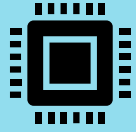
void printWifiData() {
  IPAddress ip = WiFi.localIP();
  Serial.print("IP Address: ");
  Serial.println(ip);
}
```



3-3-f Coding

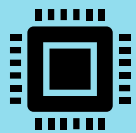
```
void loop()
{

  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  float duration_us = pulseIn(echoPin, HIGH); //Measure the integral time of pulse triging and receiving.
  float distance_cm = 0.017 * duration_us; //Change the integral into distance.
  Serial.print("distance: ");
  Serial.print(distance_cm);
  Serial.println(" cm");
  delay(500);
  if(isnan(duration_us) ){
    Serial.println("Failed to read from sensor!");
    return;
  }
}
```

3-3-g Coding

```
if (distance_cm <= 50) {
  String message = " <Warning>: The patient is getting out of bed!"; //The content of line message.
  message += "\n The distance between patient and bed head=" + String(((float)distance_cm)) + " cm";
  Serial.println(message);
  if (client1.connect(host, 443)) {
    int LEN = message.length();
    String url = "/api/notify"; //POST header
    client1.println("POST " + url + " HTTP/1.1");
    client1.print("Host: "); client1.println(host);
    //Access token
    client1.print("Authorization: Bearer "); client1.println(Linetoken);
    client1.println("Content-Type: application/x-www-form-urlencoded");
    client1.print("Content-Length: "); client1.println( String((LEN + 8)) );
    client1.println();
    client1.print("message="); client1.println(message);
    client1.println();
    delay(2000);
    String response = client1.readString();
    Serial.println(response); //Display the result of responding
    client1.stop(); //Disconnecting
  }
  else {
    Serial.println("connected fail");
  }
}
delay(5000);
if(!client.connected()){
  reconnect();
  delay(2000);
}
client.disconnect();
client.loop();
delay(300);
publishData(publishTopicStr1,distance_cm);
client.loop();
}
```



➤ 3-4-a Test MQTT Connectivity

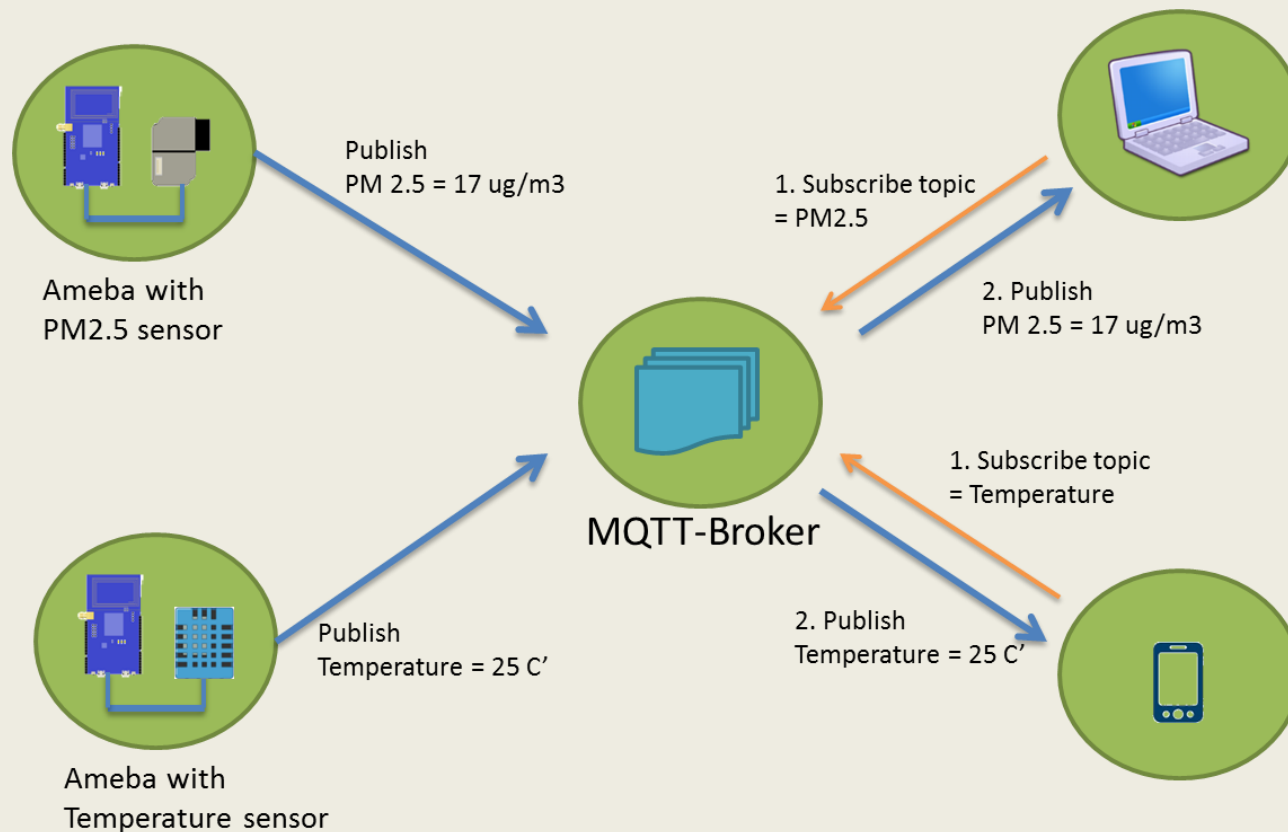
The tool is used to test MQTT server. Therefore, we can know whether the server works or not.

Subscribe

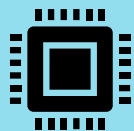
Platform subscribe the MQTT message from server.

Publish

DSI5168 publish the MQTT message. Note that the message should be sent in Json form.



(Image Source : amebaiot.com)



➤ 3-4-b Test MQTT Connectivity

Step 1

Open the Google Chrome Web store.

Step 2

Find “MQTTbox” and install it.



首頁 > 應用程式 > MQTTBox



MQTTBox

來源網站: workswithweb.com

★★★★★ 28

擴充功能

👤 30,000+ 位使用者

總覽

評論

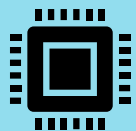
支援

相關項目

client 4 error
ws://iot.eclipse.org:80/ws
Connection Error

client 5
ws://iot.eclipse.org:80/ws
Not Connected

Menu			MQTT LOAD	Start Load Test	View Graph	View Data
Load test 1 - ws://iot.eclipse.org:80/ws, Test Type: Publishing, Msg Count: 20, Instances: 4, Topic: test			Name: Instance 1	Status: Done	Published Time: 4.7530s	
			Published Messages: 20	QoS Response: 20	QoS Time: 4.8520s	
			Load test completed successfully	Sep-28-2016 03:22:41:904 PM	🟢	
			Connection to broker closed	Sep-28-2016 03:22:41:890 PM	🟢	
			Saving data...	Sep-28-2016 03:22:41:887 PM	🟢	
			Waiting for QoS responses...	Sep-28-2016 03:22:41:767 PM	🟢	
			Publishing completed	Sep-28-2016 03:22:41:767 PM	🟢	
			Publishing messages to topic...	Sep-28-2016 03:22:37:034 PM	🟢	
			Connected to broker	Sep-28-2016 03:22:37:034 PM	🟢	
			Connecting to Broker...	Sep-28-2016 03:22:36:826 PM	🟢	



> 3-4-c MQTTbox Setting

MQTTBox Edit Help

Menu MQTT CLIENT SETTINGS

MQTT Client Name <input type="text" value="Ultrasound"/>	MQTT Client Id <input type="text" value="830346d0-896e-11eb-8e26-2532a0ef1bf0"/>	Append timestamp to MQTT client id? <input checked="" type="checkbox"/> Yes	Broker is MQTT v3.1.1 compliant? <input checked="" type="checkbox"/> Yes
Protocol <input type="text" value="mqtt / tcp"/>	Host <input type="text" value="iiot.ideaschain.com.tw"/>	Clean Session? <input checked="" type="checkbox"/> Yes	Auto connect on app launch? <input checked="" type="checkbox"/> Yes
Username <input type="text" value="xfJsQcKkZWGG8mgd05YE"/>	Password <input type="text" value="Password"/>	Reschedule Pings? <input checked="" type="checkbox"/> Yes	Queue outgoing QoS zero messages? <input checked="" type="checkbox"/> Yes
Reconnect Period (milliseconds) <input type="text" value="1000"/>	Connect Timeout (milliseconds) <input type="text" value="2000"/>	KeepAlive (seconds) <input type="text" value="5"/>	
Will - Topic <input type="text" value="v1/devices/me/telemetry"/>	Will - QoS <input type="text" value="0 - Almost Once"/>	Will - Retain <input type="checkbox"/> No	Will - Payload <input on\":1}"="" type="text" value="{\"/>

Save Delete

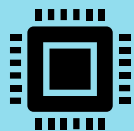
MQTT Client Name: Type the project name.

Username: device access token(refer to 3-5-e)

MQTT Client Id: Give it a unique name.

Host: <https://ideaschain.com.tw>

(Use “IDEAS Chain” as Server and platform)



➤ 3-4-d MQTTbox Testing

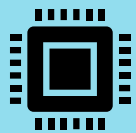
The screenshot displays the MQTTBox web application interface. At the top, there's a navigation bar with 'MQTTBox', 'Edit', and 'Help' links. Below this is a status bar showing 'Connected' with a green indicator, and buttons for 'Add publisher' and 'Add subscriber'. The main interface is divided into two panels. The left panel, titled 'Topic to publish', contains a text input field with 'v1/devices/me/telemetry', a 'QoS' dropdown set to '0 - Almost Once', a 'Retain' checkbox, a 'Payload Type' dropdown set to 'Strings / JSON / XML / Characters', and a 'Payload' text area containing the JSON string '{\"distance\":30}'. A 'Publish' button is at the bottom. The right panel, titled 'Topic to subscribe', has a text input field with 'v1/devices/me/telemetry', a 'QoS' dropdown set to '0 - Almost Once', and a 'Subscribe' button. Below the 'Subscribe' button, a text area displays the received JSON payload: '{\"distance\":{\"ts\":1617887661806,\"value\":30}}'. Below this, a detailed log shows the message details: 'qos : 0, retain : false, cmd : publish, dup : false, topic : v1/devices/me/telemetry, messageid : , length : 69, Raw payload : 12334100105115116971109910134581233411611534584954495556565554544956485444341189710811710134585148125125'.

Topic: “v1/devices/me/telemetry” It’s a specific path for IDEAS Chain.

Payload: The message should be sent in Json form.

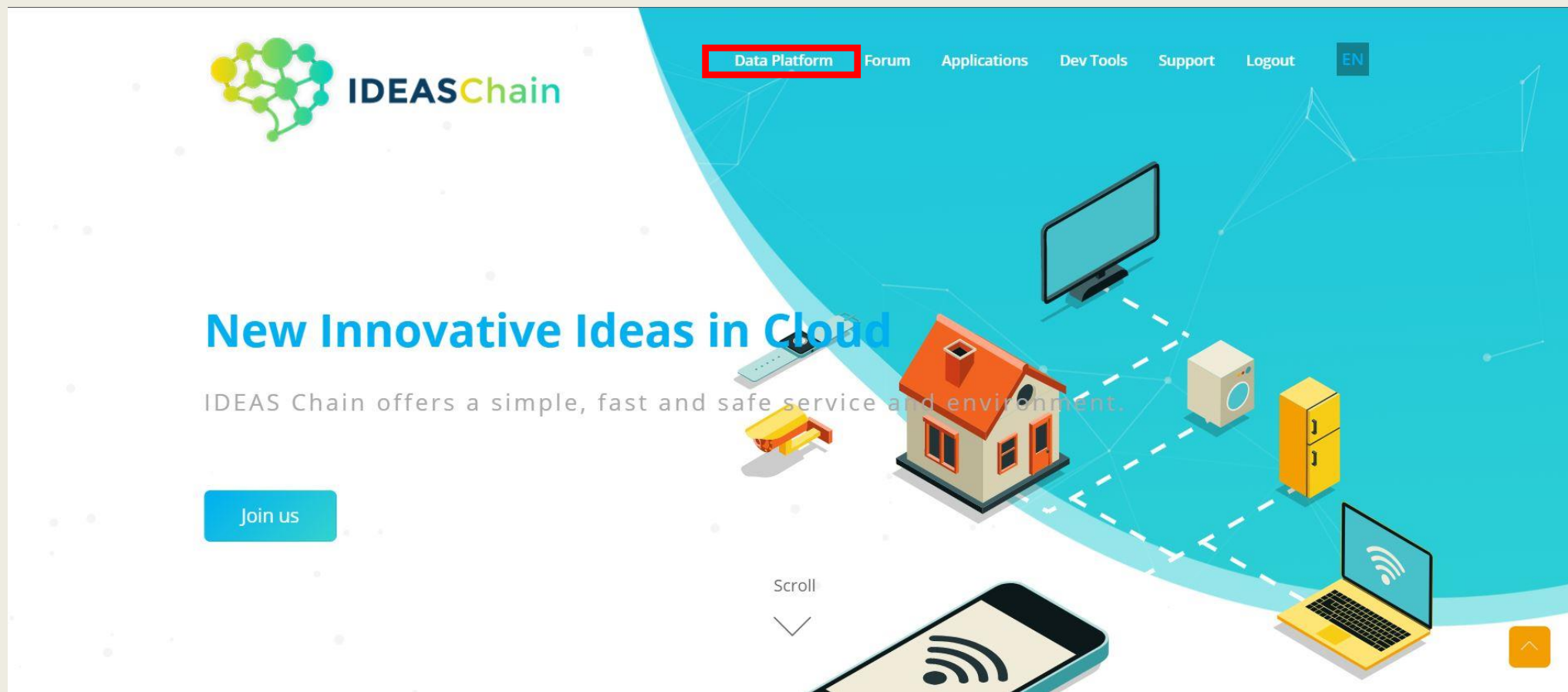
Click “Subscribe”, and click “Publish” afterwards.

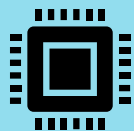
If you can successfully receive the payload, it means that the server is working normally.



➤ 3-5-a MQTT Platform

Step1. Click on the following URL" <https://iiot.ideaschain.com.tw/home> "
Open IDEAS Chain and click on "Data Platform"

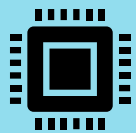




3-5-b MQTT Platform

Step2. Click on "ASSETS" on the left side.
Then, follow the steps below.

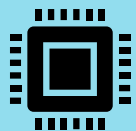
The screenshot displays the IDEASChain MQTT Platform interface. On the left sidebar, the 'ASSETS' menu item is highlighted with a red box and a red circle containing the number 1. The main content area shows a list of assets, with 'Ultrasound' selected. A modal dialog box titled 'Add Asset' is open in the center. Inside the dialog, the 'Name' field contains 'Ultrasound' and is highlighted with a red box and a red circle containing the number 3. The 'Asset type' field contains 'sensor' and is highlighted with a red box and a red circle containing the number 4. The 'Description' field is empty. At the bottom of the dialog, there are 'ADD' and 'CANCEL' buttons. In the bottom right corner of the main interface, there is a red circle containing the number 2 next to a red box containing a plus sign (+) button.



3-5-d MQTT Platform

Step3. Click on "DEVICES" on the left side.
Then, follow the steps below.

The screenshot displays the IDEASChain MQTT Platform interface. On the left sidebar, the 'DEVICES' menu item is highlighted with a red box and a red circle containing the number 1. In the main content area, the 'Add Device' dialog box is open. The dialog box has a title bar 'Add Device' with a close button. Inside the dialog, the 'Name*' field is filled with 'Ultrasound' and is highlighted with a red box and a red circle containing the number 3. The 'Device type*' field is filled with 'sensor' and has a red circle containing the number 4. Below this, there is a checkbox for 'Is gateway' which is unchecked. At the bottom of the dialog, there is a blue 'ADD' button and a 'CANCEL' button. In the bottom right corner of the main interface, there is a red circle containing the number 2, which points to a red box containing a plus sign (+) button.



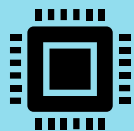
➤ 3-5-e MQTT Platform

Step4. Click on "DEVICES" on the left side.

Then, follow the steps below and paste the "ACCESS TOKEN" into the code.

The screenshot displays the IDEASChain MQTT Platform interface. On the left sidebar, the "DEVICES" menu item is highlighted with a red box and a red circle containing the number 1. The main content area shows a list of devices, with "Ultrasound" selected and highlighted with a red box and a red circle containing the number 2. Below the device name, the type "SENSOR" and status "Public" are visible. A row of action icons is shown below the device details. On the right, the "ULTRASOUND" device details page is open. The "DETAILS" tab is selected. At the top of the details page, there are three buttons: "MAKE DEVICE PRIVATE", "MANAGE CREDENTIALS", and "DELETE DEVICE". Below these, there are two buttons: "COPY DEVICE ID" and "COPY ACCESS TOKEN". The "COPY ACCESS TOKEN" button is highlighted with a red box and a red circle containing the number 3. The device details form shows the following information:

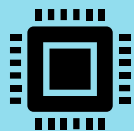
- Device is public
- Name *: Ultrasound
- Device type *: sensor
- Is gateway: ☐
- Description:



3-5-f MQTT Platform

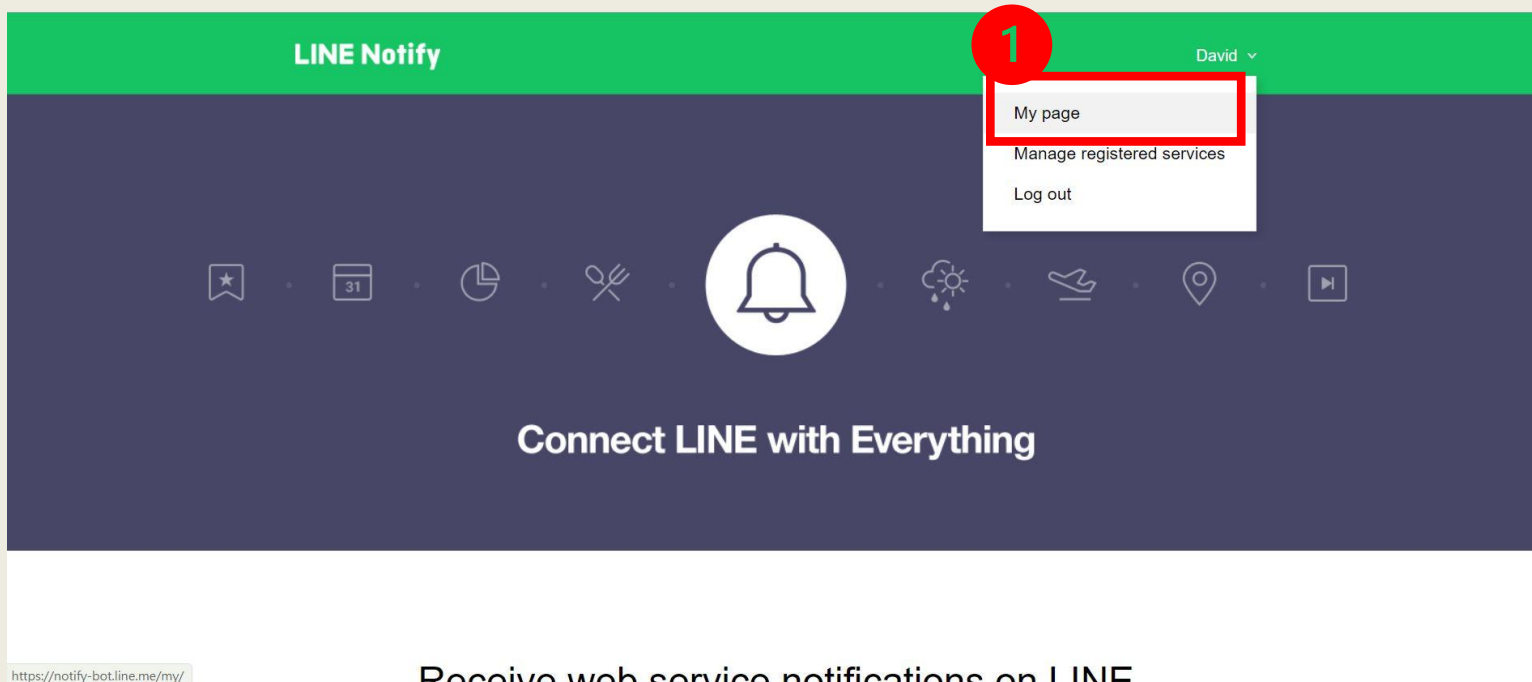
Step5. Click on "DEVICES" on the left side.
Then, follow the steps below and add the relation.

The screenshot displays the IDEASChain MQTT Platform interface. On the left sidebar, the 'DEVICES' menu item is highlighted with a red box and a red circle containing the number 1. In the main content area, the 'Devices' tab is active, and the 'Ultrasound' device is selected, indicated by a red box and a red circle containing a question mark. A modal dialog box titled 'Add relation' is open in the center. The dialog box contains the following elements: a red box around the 'Relation type' dropdown set to 'Contains' (marked with a red circle 4); a red box around the 'To entity' section where 'Device' is selected in the type dropdown and 'Ultrasound' is entered in the text field (marked with a red circle 5); and a red box around the 'Additional info (JSON)' text area. At the bottom of the dialog, the 'ADD' button is highlighted with a red box and a red circle containing the number 5. In the top right corner of the main interface, the 'RELATIONS' tab is highlighted with a red box and a red circle containing the number 3. The top navigation bar includes links for Forum, Applications, Dev Tools, and Support, along with a search icon and a user profile icon labeled 'Tenant administrator'.



> 3-6 Line Notify

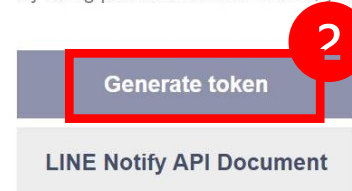
Step1. Go to the website of Line Notify . After you sign in with your account, click "My page" .

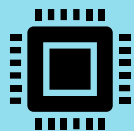


Step2. Click "Generate token" and paste it into your program.

Generate access token (For developers)

By using personal access tokens, you can configure notifications without having to add a web service.





3-5-g MQTT Platform

Step6. Click on "DEVICES" on the left side.

Then, follow the steps below and you can check the data in "DASHBOARDS".

The screenshot shows the IDEASChain MQTT Platform interface. On the left sidebar, the 'DEVICES' menu item is highlighted with a red box and a red circle with the number '1'. The main content area displays the 'ULTRASOUND' device details. The 'LATEST TELEMETRY' tab is selected, showing a table of telemetry data. A red box highlights the last row of the table, which is selected, and a red circle with the number '2' is next to it. A red circle with the number '3' is next to the 'SHOW ON WIDGET' button.

IDEASChain

Devices

ULTRASOUND
Device details

1

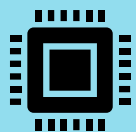
2

3

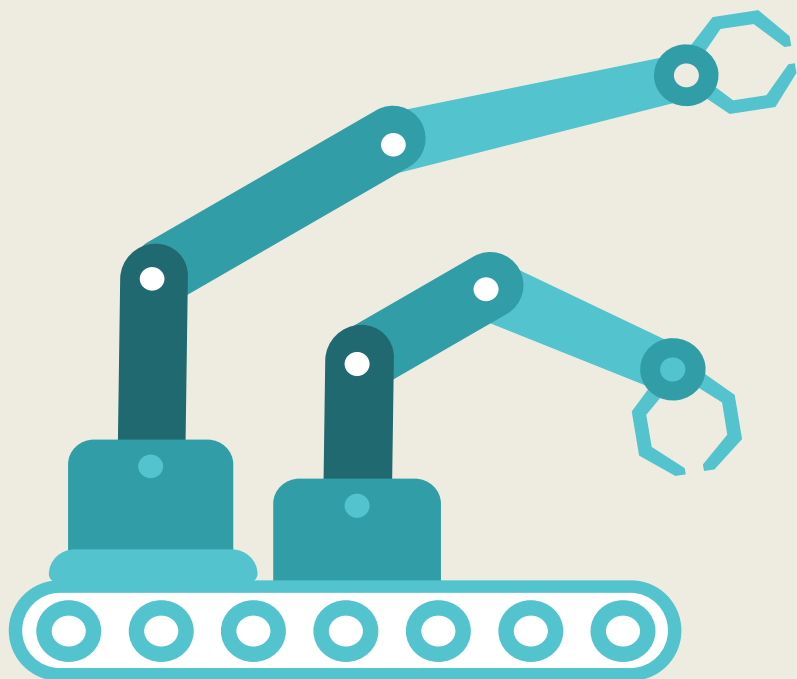
SHOW ON WIDGET

<input type="checkbox"/>	Last update time	Key	Value
<input type="checkbox"/>	2021-04-28 20:16:12	distance	30
<input type="checkbox"/>	2021-04-08 21:13:28	hello	world
<input type="checkbox"/>	2021-04-08 21:10:16	off	2
<input type="checkbox"/>	2021-04-28 21:18:57	on	1
<input checked="" type="checkbox"/>	2021-04-15 18:30:17	sensorDist	29.05

Page: 1 Rows per page: 5 1 - 5 of 6



> Chapter 4 Demonstration



4-1

WiFi Connection - Monitoring of Serial Port

4-2

MQTT Connection - Monitoring of Serial Port

4-3

Model of this Project

4-4

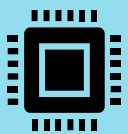
Situational simulation

4-5

IDEAS Chain dashboard

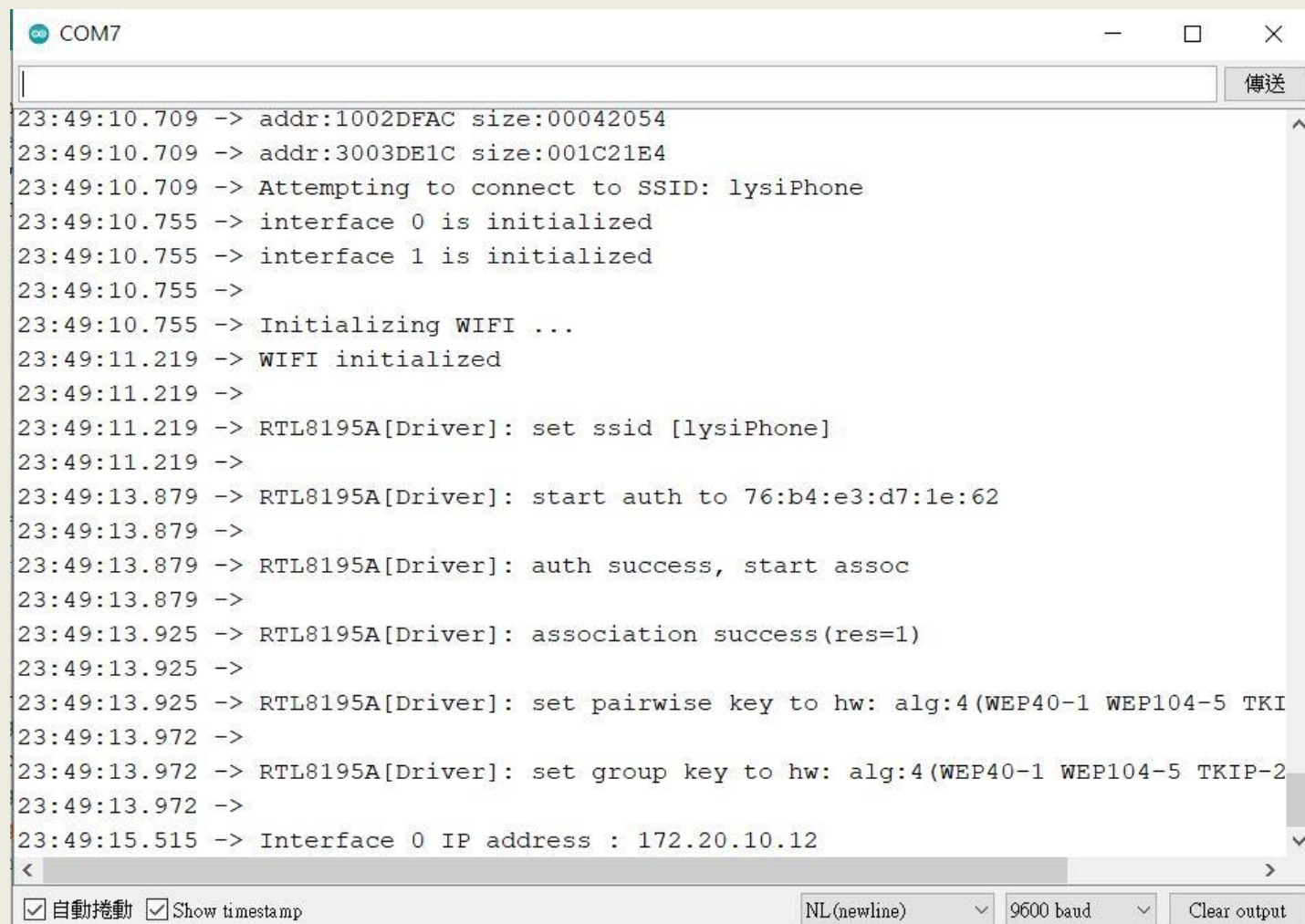
4-6

LINE Notify

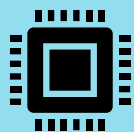


> 4-1 WiFi Connection – Monitoring of Serial Port

Verify and upload the code in Arduino IDE. And push the RST button on DSI5168, then you can view the connecting status of Wi-Fi in the Serial Monitor.



```
COM7
23:49:10.709 -> addr:1002DFAC size:00042054
23:49:10.709 -> addr:3003DE1C size:001C21E4
23:49:10.709 -> Attempting to connect to SSID: lysiPhone
23:49:10.755 -> interface 0 is initialized
23:49:10.755 -> interface 1 is initialized
23:49:10.755 ->
23:49:10.755 -> Initializing WIFI ...
23:49:11.219 -> WIFI initialized
23:49:11.219 ->
23:49:11.219 -> RTL8195A[Driver]: set ssid [lysiPhone]
23:49:11.219 ->
23:49:13.879 -> RTL8195A[Driver]: start auth to 76:b4:e3:d7:1e:62
23:49:13.879 ->
23:49:13.879 -> RTL8195A[Driver]: auth success, start assoc
23:49:13.879 ->
23:49:13.925 -> RTL8195A[Driver]: association success(res=1)
23:49:13.925 ->
23:49:13.925 -> RTL8195A[Driver]: set pairwise key to hw: alg:4(WEP40-1 WEP104-5 TKIP
23:49:13.972 ->
23:49:13.972 -> RTL8195A[Driver]: set group key to hw: alg:4(WEP40-1 WEP104-5 TKIP-2
23:49:13.972 ->
23:49:15.515 -> Interface 0 IP address : 172.20.10.12
< >
☒ 自動捲動 ☒ Show timestamp NL(newline) 9600 baud Clear output
```



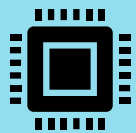
➤ 4-2 MQTT Connection - Monitoring of Serial Port

After DSI5168 has connected Wi-Fi successfully, you can check the MQTT connection status at the same Serial Monitor. Also, you can see the distance detected by the sensor afterward.

```
COM7
23:49:51.840 -> Connect to Server successful!
23:49:51.887 -> MQTT connected
23:49:52.351 -> distance: 203.37 cm
23:49:53.188 -> {"sensorDist ":"203.37 "}
23:49:53.188 -> Attempting MQTT connection Attempt to connect...
23:49:53.235 ->
23:49:53.235 -> Connect to Server successful!
23:49:53.235 -> MQTT connected
23:49:53.421 -> distance: 207.52 cm
23:49:54.298 -> {"sensorDist ":"207.52 "}
23:49:54.298 -> Attempting MQTT connection Attempt to connect...
23:49:54.345 ->
23:49:54.345 -> Connect to Server successful!
23:49:54.345 -> MQTT connected
23:49:54.481 -> distance: 164.46 cm
23:49:55.362 -> {"sensorDist ":"164.46 "}
23:49:55.362 -> Attempting MQTT connection Attempt to connect...
23:49:55.408 ->
23:49:55.408 -> Connect to Server successful!
23:49:55.408 -> MQTT connected
23:49:55.548 -> distance: 94.42 cm
```

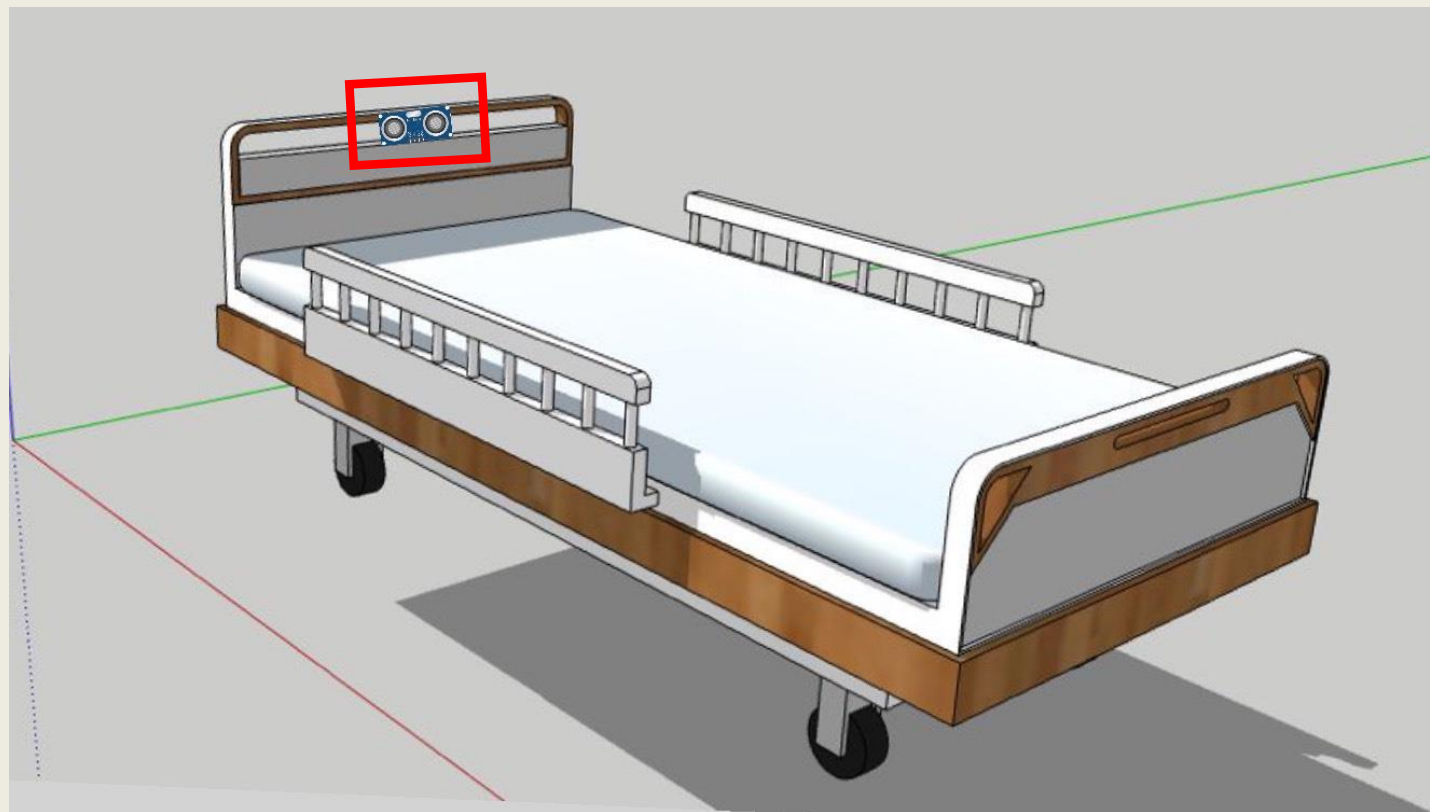
傳送

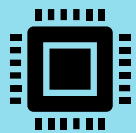
☒ 自動捲動 ☒ Show timestamp NL(newline) 9600 baud Clear output



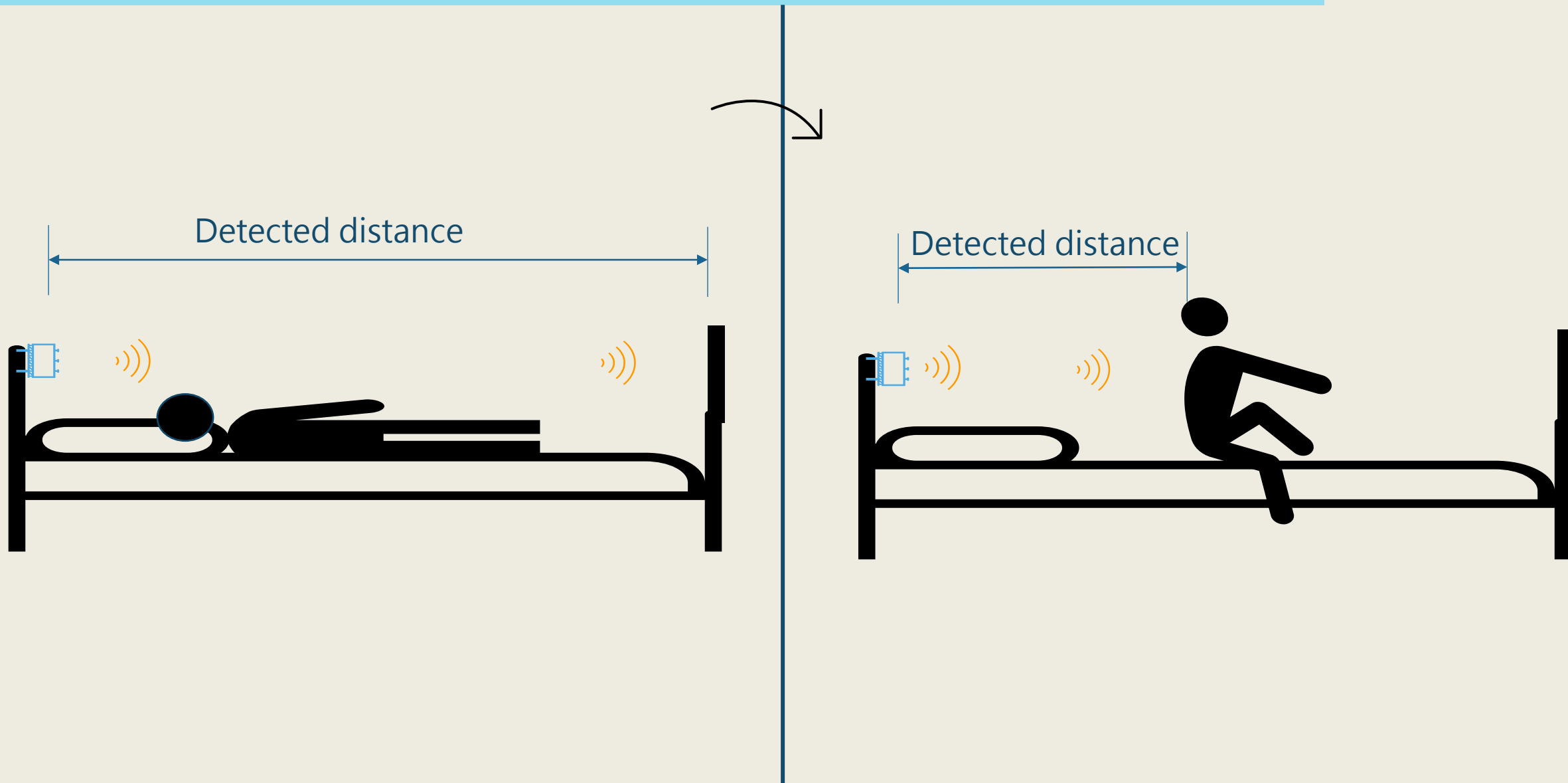
➤ 4-3 Model of this Project

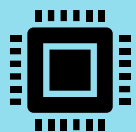
In this project, I set the Ultrasonic sensor in front of the bed. If the bedridden gets up, the distance detected by the sensor will reduce. So that, we can get the latest information about the bedridden.



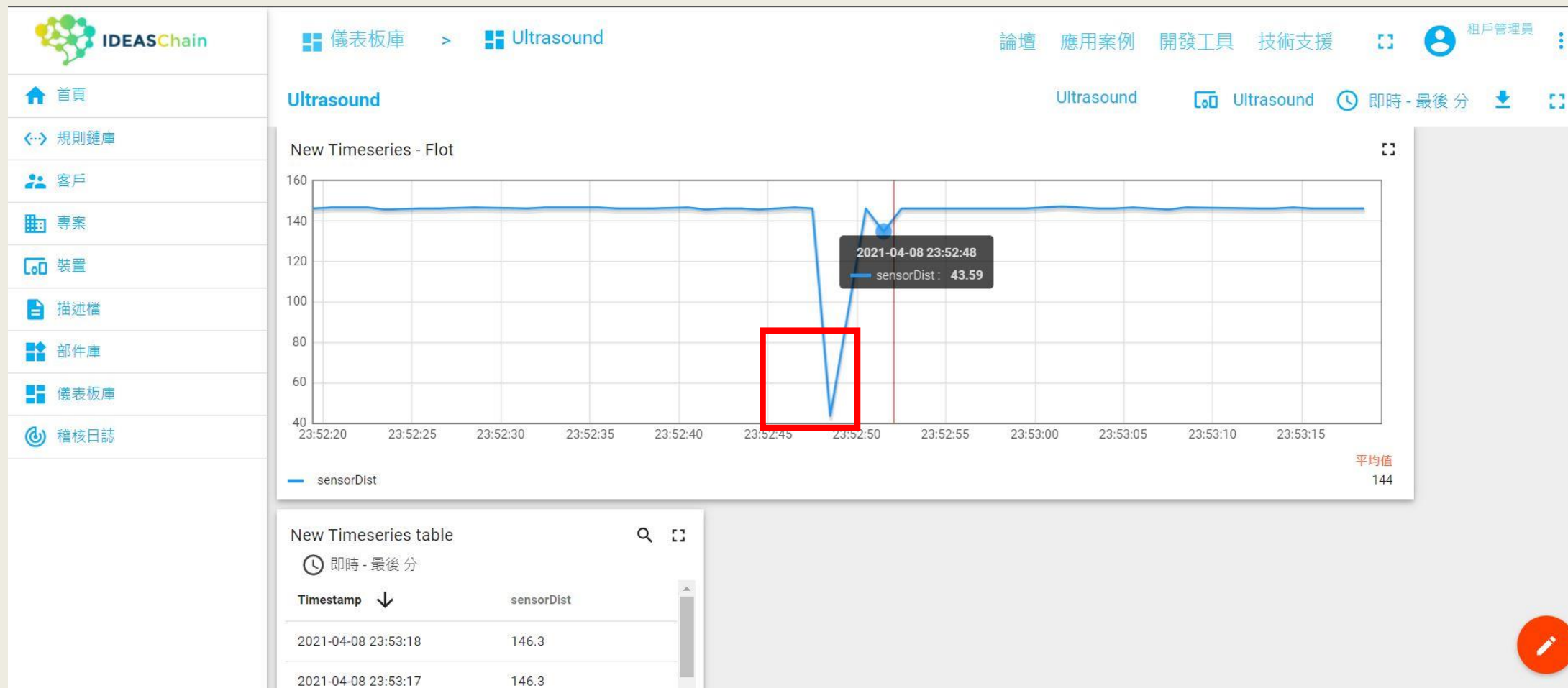


➤ 4-4 Situational simulation

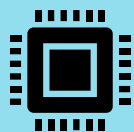




> 4-5 Ideaschain dashboard

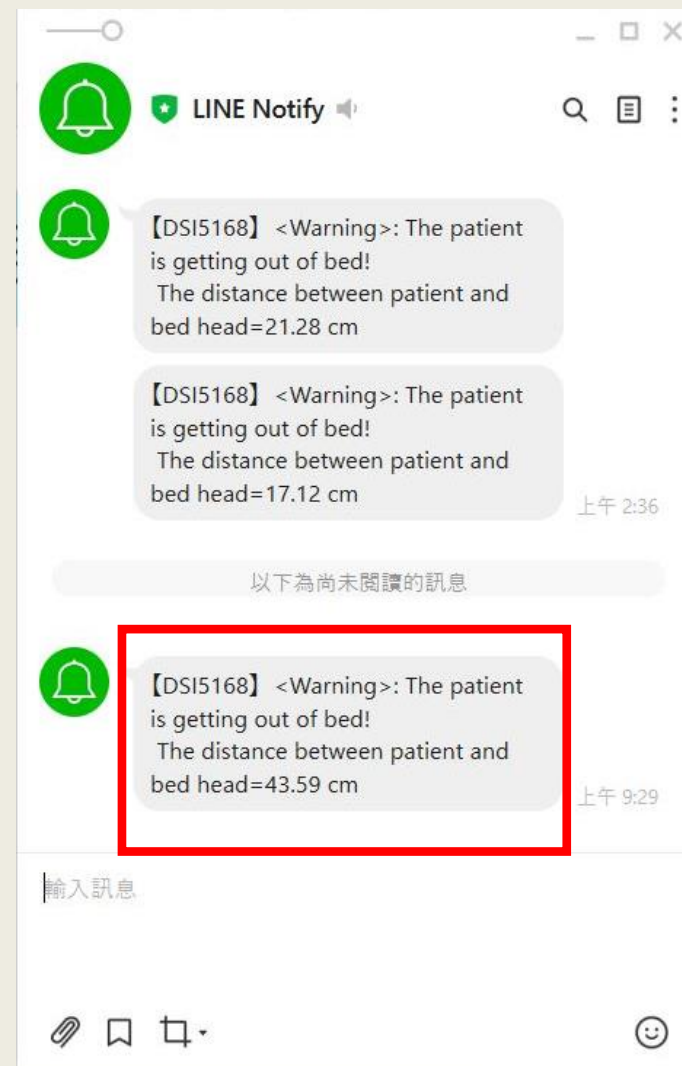


Open the page of the "DASHBOARDS" in IDEAS Chain, and you can get the latest information posted by DSI5168. If the distance decrease in a sudden, it means that the bedridden is getting up.



> 4-6 LINE Notify

If the detected value decreases in a sudden, until it lower than 50cm, the dev board will send an alarm message to LINE Notify by using SSL protocol. Therefore, even if the caregivers are not on-site, they can get the notification by LINE Notify.



<< Thank you >>

