



AgroSense_Pipe Pressure Sensor

LoRaWAN® Manual

V1.0

Author: Yuki

Time: 2025.01.08

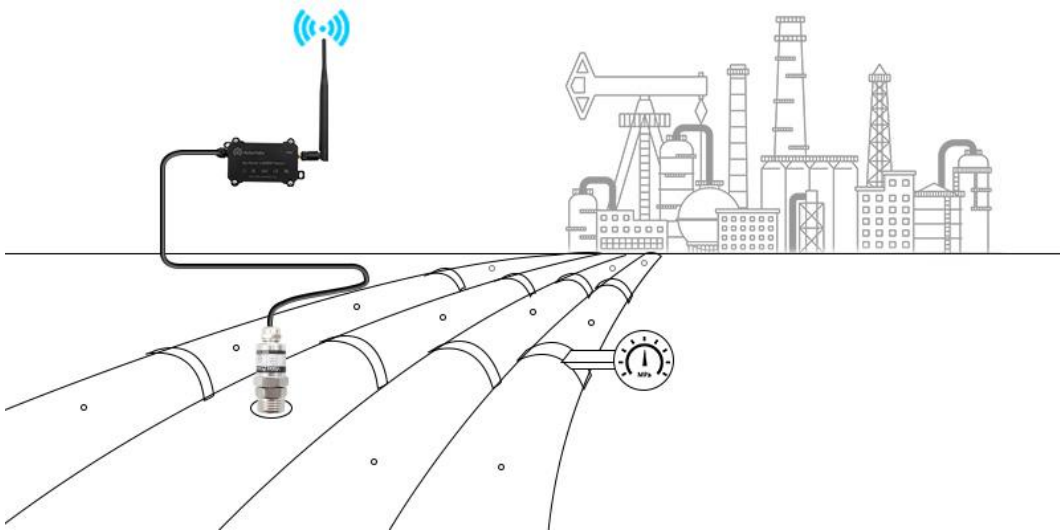
Contents

1 Product Description	1
1.1 Introduction	1
1.2 Feature	2
1.3 Parameter	2
2 Technical route	4
2.1 System Framework	4
2.2 Regional frequency band	4
3 Usage	6
3.1 TTN and ThingSpeak	6
3.1.1 Network Server configuration	6
3.1.2 Decoder	9
3.1.3 Application Server configuration	11
3.1.4 Connect the Network Server and Application Server	12
3.1.5 Downlink	13
3.2 Datacake	15
3.2.1 Downlink	22

1 Product Description

1.1 Introduction

This AgroSense LoRaWAN® pressure Sensor detects water/air pressure in the pipe, and thus to check if there are any blocking/frozen. This sensor reports the data to TTN/DataCake via LoRaWAN. It also stores max 3K results in internal flash so if LoRaWAN connection temporary not available temporary for some reason it can store the data and resend them as LoRaWAN recover.



Benefits from LoRaWAN®, which ensures stability and reliability. It is capable of covering a long transmission range while maintaining low power consumption. Unlike wireline devices, it is battery-powered, reducing the workload and complexity of deployment, design and development for end-users that can work via powering it, and setting the configuration in the cloud server.



1.2 Feature

- Includes a **high precision** sensor.
- Compatible with Worldwide **LoRaWAN® Networks**: Support the universal frequency bands EU868/ US915.
- LoRaWAN version: LoRaWAN Specification **1.0.3**.
- **Long Range**: Up to 2 kilometers in the city, up to 10 kilometers in the wilderness, receive sensitivity -137dBm , transmit power up to 21dBm.
- **Ultra-low power** consumption design, traditional AAA alkaline dry battery can be used for one year.
- **Data encryption**: Provide end-to-end secure communication, including device authentication and network data encryption, to ensure the security of data transmission and prevent data theft and malicious attacks.
- **High stability and reliability**: good stability in noisy environments, able to penetrate buildings and obstacles, so it can maintain good communication quality in urban and suburban environments.
- Suitable for **Harsh Environments**: Can work normally under the temperature of -40℃ ~ 85℃, IP68 waterproof, suitable for outdoor use in harsh conditions, high UV, dusty, heavy rain and other bad weather.
- Monitor data and upload **real-time** data regularly.
- Modify the product parameters through **AT commands**.
- Support **downlink** to modify the time interval, motion status on/off, motion status sensitivity.

1.3 Parameter

1. General Parameters

Product Model	AGLWPP01
Pressure Type	Non-corrosive liquid or gas
Range	0~1.6 Mpa
Accuracy	± 0.5% FS
Resolution	1 kPa (0.01 Bar)
Overload	3.0 Mpa
Long-term Stability	± 0.3% FS/year
Process Connection	G1/2
Operating Temperature	-20~85℃

Cycle Life	10 million times
------------	------------------

2.Wireless Parameters

Communication Protocol	Standard LoRaWAN® protocol V1.0.3
Network Access/Operating Mode	OTAA Class A
MAX Transmit Power	21dBm
Receiver Sensitivity	-137dBm/125kHz SF=12
Frequency Band	EU868/US915

3.Physical Parameters

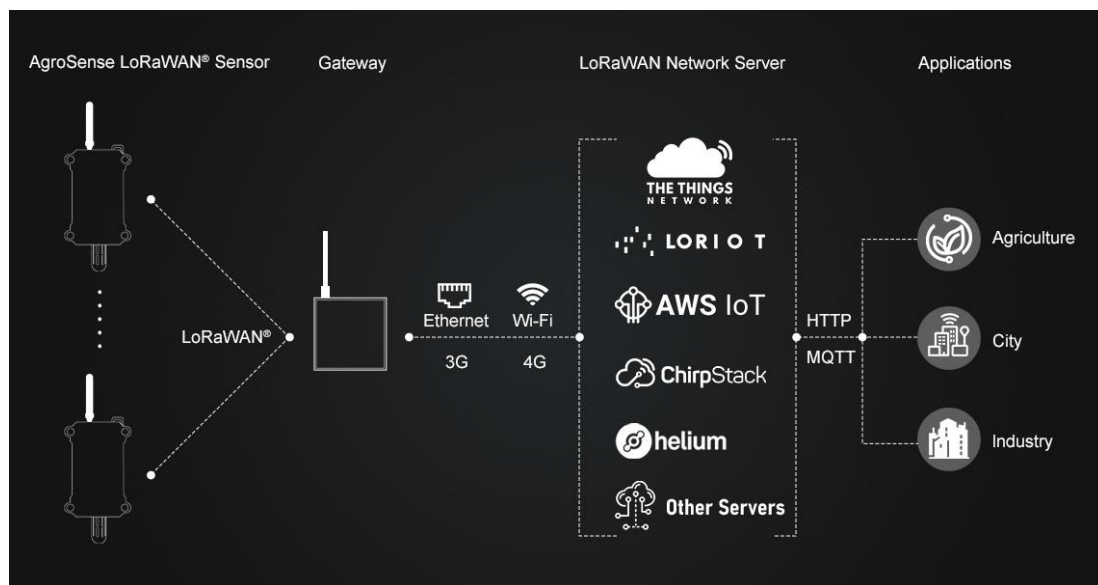
Lead Length	1 meter
Power Supply	1 x 18650 3.7V Lion batteries
Operating Temperature	-40°C ~85°C
Protection Class	IP68
Dimensions	131 × 62.7 × 27.5 mm
Mounting	Wall Mounting

2 Technical route

2.1 System Framework

AgroSense Pipe Pressure Sensor uses LoRAWAN technology, and its network architecture includes four parts: End Nodes, Concentrator/Gateway, Network Server and Application Server.

End Nodes	It is responsible for collecting sensing data and then transmitting it to Gateway via the LoRaMAC protocol.
Concentrator/Gateway	It is mainly responsible for transmitting node data to the server.
Network Server	Organize the data into JSON packets and decode them.
Application Server	Display the data.



The steps to achieve the detection of pressure is:

1. Collect the pressure data by sensor, and send the data from End Node to Gateway.
2. The Gateway packages node data and transmits it to the Network Server.
3. The Network Server decodes the data and sends it to the Applications.
4. Finally, user can monitor the pressure in the APP.

2.2 Regional frequency band

At the present moment, our product solely accommodates compatibility with the US915 and

EU868.

area	frequency band	center frequency
China	470-510MHz	CN486MHz
America	902-928MHz	US915MHz
Europe	863-870MHz	EU868MHz
Korea	920-923MHz	KR922MHz
Australia	915-928MHz	AU923MHz
New Zealand	921-928MHz	NZ922MHz
Asia	920-923MHz	AS923MHz

3 Usage

We use The Things Network as our Network Server, we need to configuration the country/ area frequency, inputting DEV EUI/ APP EUI/ APP Key, decodes, and connect to ThingSpeak or Datacake.

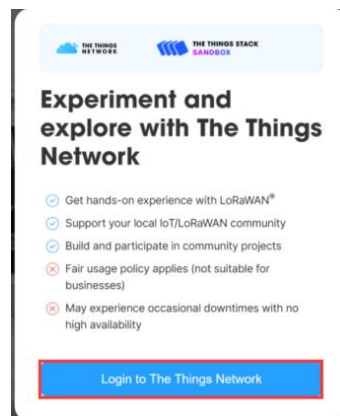
DEV EUI	Unique identification of device, authorized by IEEE
APP EUI	Unique identification of application
APP Key	One of the join network parameters on OTAA mode, calculated by DE EUI

- End Nodes and Gateway: AgroSense Pipe Pressure Sensor LoRaWAN®. (The AgroSense series is applicable)
- Network Server: The Things Network. (Loriot, AWS IoT, ChirpStack, ect)
- Application Server: ThingSpeak.(Datacake, Blockbax, akenza, ect)

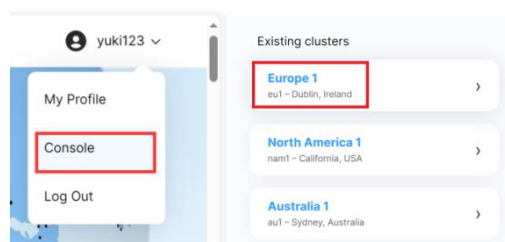
3.1 TTN and ThingSpeak

3.1.1 Network Server configuration

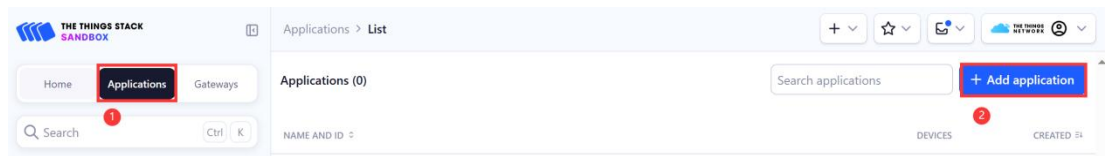
- Open The Things Network in your browser and login it. (Or register an account)



- Click “Console” and select clusters. (we take the European region for example.)



- Click “Go to applications” --> “+ Create application”.



- Write the Application ID and click “Create application”.

Application ID *

agrosense-sensor

Application name

My new application

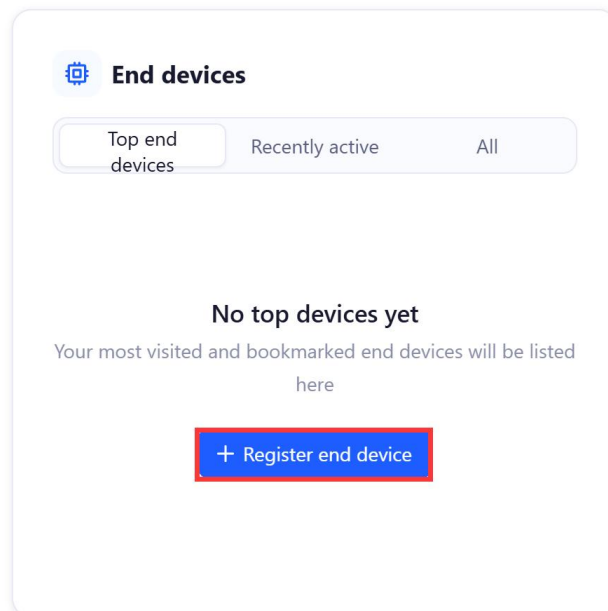
Description

Description for my new application

Optional application description; can also be used to save notes about the :

Create application

- Click “+ Register and device”.



- Following the steps, and input the DEV EUI/ APP EUI/ APP Key (notice: JoinEUI=APP EUI) and subsequently click on "Register end device" to complete the registration process.

End device type

Input method ⓘ

☐ Select the end device in the LoRaWAN Device Repository

☒ Enter end device specifics manually 1

Frequency plan ⓘ *

Europe 863-870 MHz (SF9 for RX2 - recommended) | v

LoRaWAN version ⓘ *

LoRaWAN Specification 1.0.3 | v

Regional Parameters version ⓘ *

RP001 Regional Parameters 1.0.3 revision A | v 2

[Show advanced activation, LoRaWAN class and cluster settings](#)

Provisioning information

JoinEUI ⓘ *

48 FF 00 00 00 00 01 65 3 Confirm 4

Continue, please enter the JoinEUI 3 and device so we can determine onboarding options 4

Provisioning information

JoinEUI ⓘ *

48 FF 00 00 00 00 01 65 Reset

This end device can be registered on the network

DevEUI ⓘ *

48 E6 63 FF FE 30 01 65 Generate 0/50 used

AppKey ⓘ *

4A 35 62 6B 95 AB 5B 4D 3F 3B DE 12 71 B1 6F 2A Generate

End device ID ⓘ *

eui-48e663ffe300165

This value is automatically prefilled using the DevEUI

After registration

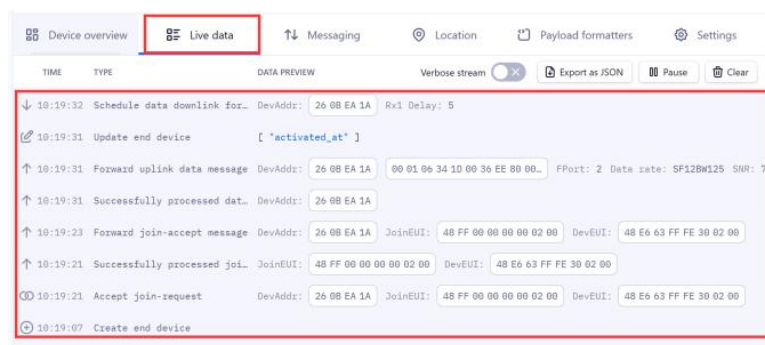
☒ View registered end device

☐ Register another end device of this type

Register end device

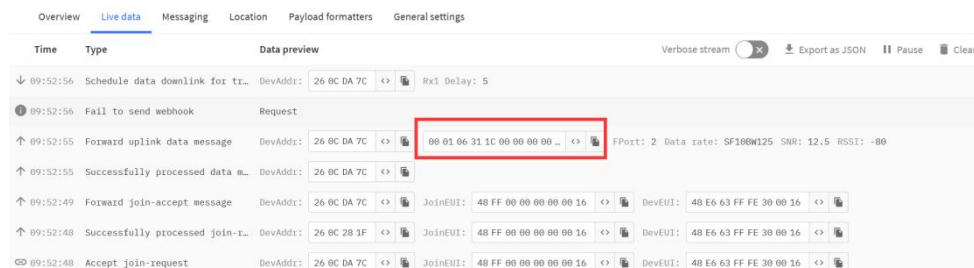


- Plug the battery and press RES button, you can see the device is connected successfully in the TTN.



3.1.2 Decoder

- Now, we need to decode the data.



Data length	Data description	Value range	Explanation
byte 0	Data packet sequence number high 8 bits	0-0xFFFF	Counting starts from 0 and increments, resetting back to 0 after reaching 65535
byte 1	Data packet sequence number low 8 bits		
byte 2	Battery voltage		The value is obtained by amplifying the data by 10 times, and the actual value needs to be divided by 10 to convert to the actual battery voltage. The purpose of multiplying by 10 is to retain one decimal place of the voltage value. For example, if the value is 0x21 = 33, then the battery voltage is 3.3V.

AgroSense_Pipe Pressure Sensor LoRaWAN®

byte 3	Pipe Pressure ADC bits 8 to 15		ADC value in millivolts For example, if the value is 0x0CE4 = 3300, then the value is 3300mV.
byte 4	Pipe Pressure ADC bits 0 to 7		
byte 5	NC		
byte 6	NC		
byte 7	NC		
byte 8	NC		
byte 9	NC		
byte 10	NC		
byte 11	NC		
byte 12	NC		
byte 13	data transmission interval bits 24 to 31	0-0xFFFFFFFF	The time interval for data transmission has been increased by a factor of 1000. The unit is seconds.
byte 14	data transmission interval bits 16 to 23		
byte 15	data transmission interval bits 8 to 15		
byte 16	data transmission interval bits 0 to 7		
byte 17	The data validity flag	0/1	0 is invalid, 1 is valid.
Fport 1	Change the data sending interval		
Fport 5	Upload the quantity of the latest local logged data		

Example: 0x00, 0x01, 0x28, 0x0C, 0xE4, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x36, 0xEE, 0x80, 0x01

Data parsing:

Battery voltage is 4V.

Pipe Pressure ADC is 3300mV. (The output is a voltage value, the pressure has to be converted in the decoder.)

Data transmission interval value is 3600s.

- Know how to decode it after, we need to write it in code. (You can check it out on [Github](#))

```
function decodeUplink(input) {
```

```

var num = input.bytes[0] * 256 + input.bytes[1]
var bat = input.bytes[2] / 10.0
var Volt_Pressure = (input.bytes[3] * 256 + input.bytes[4]) / 1000.0
var Pipe_Pressure = (Volt_Pressure-0.483)*250 //KPa
    Pipe_Pressure = Math.round(Pipe_Pressure * 1000) / 1000; // keep three decimal places
var time_interval = (input.bytes[13] * 16777216 + input.bytes[14] * 65536 + input.bytes[15] * 256 +
input.bytes[16]) / 1000.0//S

return {
    data: {
        field1: bat,
        field2: Volt_Pressure,
        field3: Pipe_Pressure,
        field4: time_interval,
    },
};
}

```


- Select “Payload formatters” and follow the steps.

The screenshot shows the TTN Application Server configuration interface. The 'Payload formatters' tab is selected and highlighted with a red box and a red circle with the number 1. Below the tabs, the 'Setup' section is visible. The 'Formatter type*' dropdown menu is set to 'Custom Javascript formatter' and is highlighted with a red box and a red circle with the number 2. The 'Formatter code*' text area contains the provided JavaScript code and is highlighted with a red box and a red circle with the number 3. At the bottom, the 'Save changes' button is highlighted with a red box.

3.1.3 Application Server configuration

In the Application Server configuration, we need to create ThingSpeak channel and get Channel ID and API Key, this is the key to our connection to TTN.

- Login to the ThingSpeak. (Or register an account)




Email

No account? [Create one!](#)

By signing in, you agree to our [privacy policy](#).

Next

- Click “New Channel”, fill in the Channel name and field names and click “Save Channel”.



My Channels

New Channel

Name: AgroSense_Pipe Pressure Sensor

Description:

Field 1: Bat ☒

Field 2: Volt_Pressure ☒

Field 3: Pipe_Pressure ☒

Field 4: Time_interval ☒

Save Channel

- After successful creation, copy the Channel ID and API Key.

Channel ID: 2601128

Author: mwa0000034232775

Access: Private

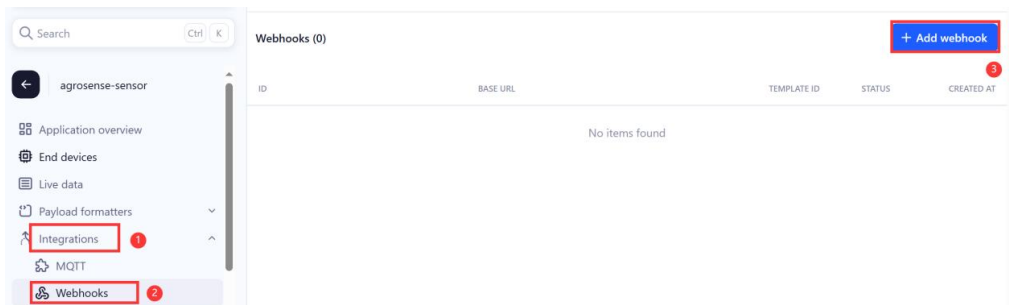
Private View Public View Channel Settings Sharing API Keys

Write API Key

Key: RXASRM3U00ACSSW1

3.1.4 Connect the Network Server and Application Server

- In the TTN, click “integrations” --> “Webhooks” --> “+ Add webhook”.



Search

agrosense-sensor

Application overview

End devices

Live data

Payload formatters

Integrations

MQTT

Webhooks

Webhooks (0)

+ Add webhook

ID

BASE URL

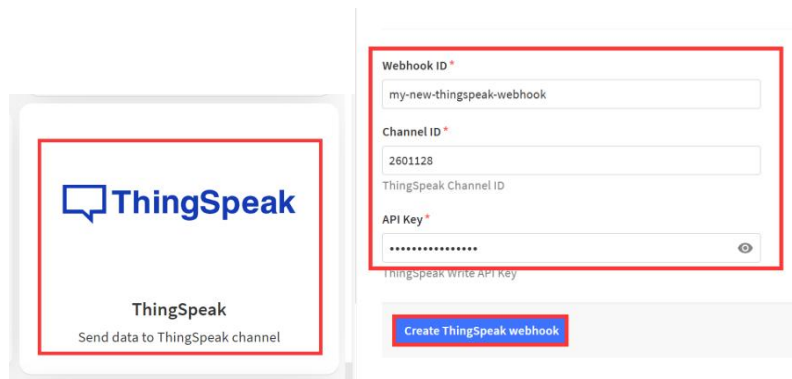
TEMPLATE ID

STATUS

CREATED AT

No items found

- Select “ThingSpeak”, Fill in the Webhook ID and paste the Channel ID and API Key, click “Create ThingSpeak Webhook”.



The screenshot shows the ThingSpeak 'Create ThingSpeak webhook' form. The form is divided into two main sections. The top section contains three input fields: 'Webhook ID *' with the value 'my-new-thingspeak-webhook', 'Channel ID *' with the value '2601128', and 'API Key *' with a masked value. The bottom section contains a 'Create ThingSpeak webhook' button. All these elements are highlighted with red rectangular boxes.

- Press RES button, wait about a minute, you will successfully see the data in ThingSpeak.(You will receive the data every hour.)



3.1.5 Downlink

The downlink has two functions:

- Modification time interval (Fport1)

Modify the time interval for uploading data, the default is one hour.

- Upload the quantity of the latest local logged data (Fport5)

Users can view previous data based on this feature.

1 、 If you need to change time Interval (Default 60 minutes), you can click “Payload formatters-->Downlink” and follow the steps.

Formatter code you can find in [Github](#).

Device overview Live data Messaging Location Payload formatters

Uplink Downlink 2

1

Setup

Formatter type* Custom Javascript formatter 3

Formatter code* 4

```

1 // Encoder function to be used in the TTN console for downlink payload
2 function Encoder(input) {
3   var minutes = input.minutes;
4
5   // Converting minutes to seconds
6   var seconds = minutes * 60;
7
8   // If the number of seconds is less than 300 seconds, set it to 300 seconds
9   if (seconds < 300) {
10    seconds = 300;
11  }
12
13  var payload = [
14    (seconds >> 24) & 0xFF,
15    (seconds >> 16) & 0xFF,
16    (seconds >> 8) & 0xFF,
17    seconds & 0xFF
18  ];
19
20  return payload;
21 }

```

2、Click “Save changes”.

Save changes

3、Click “Messaging-->Schedule downlink”.

Note: you must use this format:

```

{
  "minutes": 5
}

```

Device overview Live data Messaging

Schedule downlink Simulate uplink

Schedule downlink

Insert Mode

☒ Replace downlink queue

☐ Push to downlink queue (append)

FPort* 1

Payload type

☐ Bytes ☒ JSON

Payload

```

1 {
2   "minutes": 5
3 }

```

The decoded payload of the downlink message

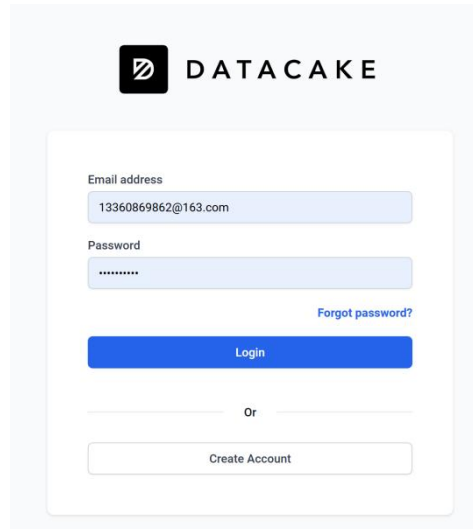
☐ Confirmed downlink

Schedule downlink

4、The modified interval will be updated after the next data upload.

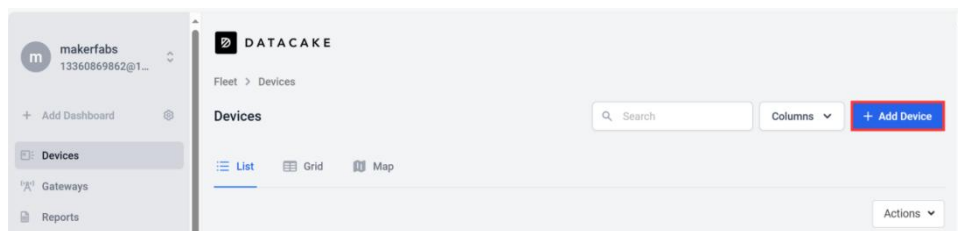
3.2 Datacake

1、Login datacake or Create Account

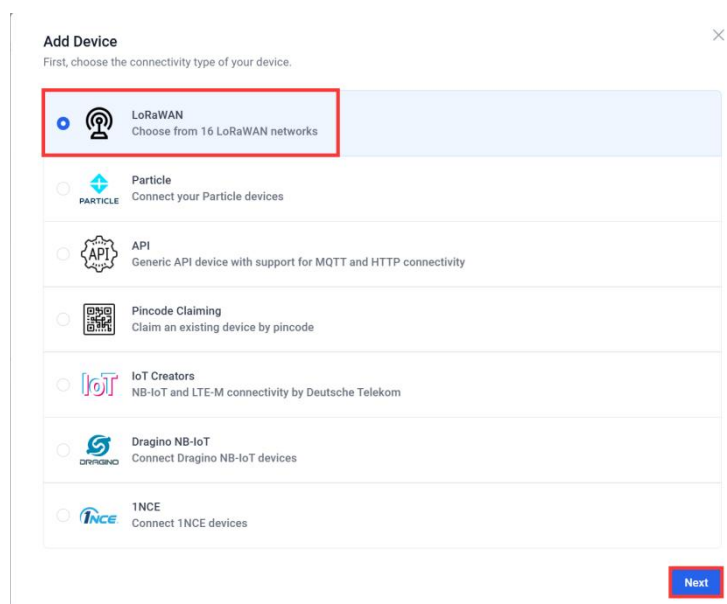


The image shows the Datacake login and registration interface. At the top is the Datacake logo. Below it is a form with two input fields: 'Email address' (containing '13360869862@163.com') and 'Password' (containing '*****'). There is a 'Forgot password?' link next to the password field. Below the password field is a blue 'Login' button. Underneath the login button is a horizontal line with the word 'Or' in the center. Below that is a white 'Create Account' button.

2、Click “Add Device”



3、Select LoRaWAN and click “Next”



The image shows the 'Add Device' dialog box. At the top is the title 'Add Device' and a close button. Below the title is the instruction 'First, choose the connectivity type of your device.' There is a list of connectivity options, each with a radio button and an icon. The 'LoRaWAN' option is selected and highlighted with a red box. The other options are: 'Particle', 'API', 'Pincode Claiming', 'IoT Creators', 'Dragino NB-IoT', and '1NCE'. At the bottom right is a blue 'Next' button highlighted with a red box.

4、Select a Product based on your needs, take "Create new empty product" as an example.

5、Select "Datacake LNS"

6、Enter DEVEUI、APPEUI、APPKEY、FREQUENCY (take 915 for example) and DEVICE CLASS.

7、Choose the type according to your needs, and click “Add 1 device”.

8、Click to go to the device you just added.

9、Click “Configuration”, enter Decoder and click “Save”.(You can check it out on [Guihub](#))

```
function Decoder(payload, port) {
    var input = {
        bytes: payload
    };

    // var num = input.bytes[0] * 256 + input.bytes[1];
    var bat = input.bytes[2] / 10.0;
    var Volt_Pressure = (input.bytes[3] * 256 + input.bytes[4]) / 1000.0
    var Pipe_Pressure = (Volt_Pressure-0.483)*250 //KPa
    Pipe_Pressure = Math.round(Pipe_Pressure * 1000) / 1000; // keep three decimal places
    var time_interval = (input.bytes[13] * 16777216 + input.bytes[14] * 65536 + input.bytes[15] * 256 +
input.bytes[16]) / 1000.0//S

    var decoded = {
        bat: bat,
        Pipe_Pressure: Pipe_Pressure,
    };

    // Test for LoRa properties in normalizedPayload
    try {
        console.log('normalizedPayload:', normalizedPayload); // Log to check normalizedPayload structure

        decoded.lora_rssi =
            (normalizedPayload.gateways    &&    Array.isArray(normalizedPayload.gateways)    &&
normalizedPayload.gateways.length > 0 && normalizedPayload.gateways[0].rssi) || 0;
        decoded.lora_snr =
            (normalizedPayload.gateways    &&    Array.isArray(normalizedPayload.gateways)    &&
normalizedPayload.gateways.length > 0 && normalizedPayload.gateways[0].snr) || 0;
        decoded.lora_datarate = normalizedPayload.data_rate || 'not retrievable';
    } catch (error) {
        console.log('Error occurred while decoding LoRa properties: ' + error);
    }

    return [
        { field: "bat", value: decoded.bat },
        { field: " Pipe_Pressure", value: decoded. Pipe_Pressure },
        { field: "lora_rssi", value: decoded.lora_rssi },
        { field: "lora_snr", value: decoded.lora_snr },
        { field: "lora_datarate", value: decoded.lora_datarate }
    ];
}
```

10、 Follow the steps to add a field. (Every fields is the same way)

Fields

Fields describe the data the device will store.

[+ Add Field](#)

Add Field

Fields define the schema of the data the device stores.

Type: Float

Name: bat

Identifier: BAT

Unit: Optional

Role: None

You can define the role of a field, which are unique per product and can be used to add context to global visualisations and reports.

Formula: Optional

Formulas can be used to perform calculations on values from other fields. Fields that have a formula cannot be written to from a decoder or via the API.

☐ Use Formula

Cancel Add Field

Fields

Fields describe the data the device will store.

NAME	IDENTIFIER	TYPE
bat	BAT	Float
Pipe_Pressure	PIPE_PRESSURE	Float

11、Press RST button, wait until the sensor connects to the gateway successfully, you will see the data the sensor is currently reading.

Fields

Fields describe the data the device will store.

[+ Add Field](#)

● Live data

NAME	IDENTIFIER	TYPE	ROLE	CURRENT VALUE	LAST UPDATE
bat	BAT	Float	N/A	4	1 minute ago
Pipe_Pressure	PIPE_PRESSURE	Float	N/A	2	1 minute ago

12、To get a better look at the data, we can add widget.

Click “Dashboard-->switch-->+ Add Widget”.

DATA CAKE

Fleet > agrosense sensor

agrosense sensor

Serial Number: 48E663FFFE30001F Last update: Never

Dashboard History Downlinks Configuration Debug Rules Permissions

Public Link + Add Widget

13、Select “Value” and set Title, Field and presentation form as well as the interval color.

The first screenshot shows the widget selection menu with the 'Value' widget highlighted. The second screenshot shows the 'Edit Value Widget' dialog with the 'Basics' tab selected. The title is set to 'Pipe_pressure' and the field is set to 'Pipe_Pressure'. The third screenshot shows the 'Gauge' tab selected, with the 'Gauge Type' set to 'Circular'. The 'Values' table is configured with 6 rows, each with a color code. The 'Unit' is set to 'kPa' and 'Decimal Places' is set to 2. The 'Save' button is highlighted.

Values	Color
0	#38b2ac
1	#48bb78
2	#ecc94b
3	#ed8936
4	#f56565
5	#ccc

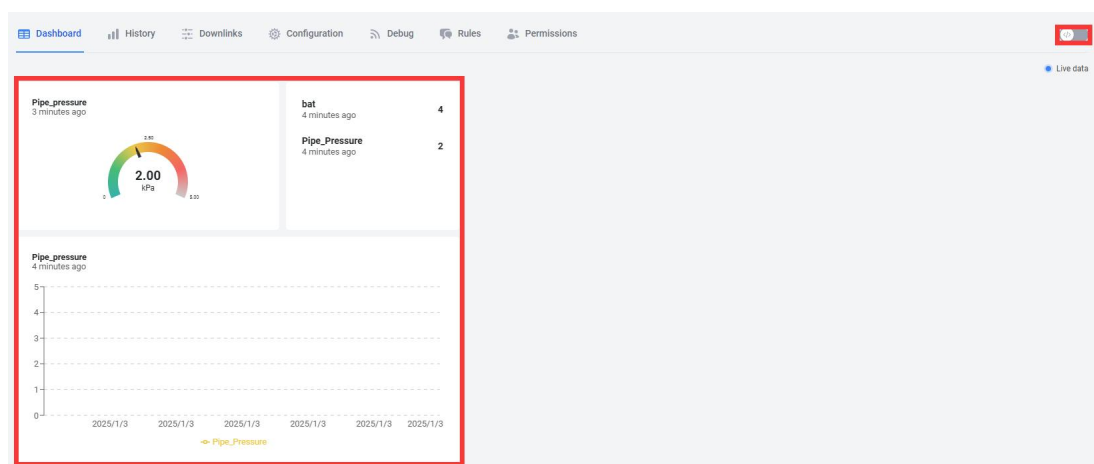
14、Select Chart and set Title, Field, Kind, Line Thickness and click “save”.

The first screenshot shows the widget selection menu with the 'Chart' widget highlighted. The second screenshot shows the 'Edit Chart Widget' dialog with the 'Basic' tab selected. The title is set to 'Pipe_pressure' and the field is set to 'Pipe_Pressure'. The 'Save' button is highlighted.

15、Select Device Fields, check “Fields” and click “Save”.

16、Click the switch to save, and you can see the data visually.

17、The steps for humidity are the same as above, and you can add your own.



3.2.1 Downlink

The downlink has two functions:

- Modification time interval (Fport1)

Modify the time interval for uploading data, the default is one hour.

- Upload the quantity of the latest local logged data (Fport5)

Users can view previous data based on this feature.

1 、 If you need to change time Interval (Default 60 minutes), you can click “Configuration-->Fields-->+Add Field”

Add Field
Fields define the schema of the data the device stores.

Type: Integer

Name: Sending Time Interval

Identifier: SENDING_TIME_INTERVAL
The field identifier is a unique string that can consist of uppercase letters, numbers and underscores. Once a field has been created, the identifier can not be changed.

Unit: Optional

Role: None
You can define the role of a field, which are unique per product and can be used to add context to global visualisations and reports.

Formula: Optional
Formulas can be used to perform calculations on values from other fields. Fields that have a formula can not be written to from a decoder or via the API.

☐ Use Formula

Cancel Add Field

2、 Click “Downlink-->Add Downlink”.

Dashboard History **Downlinks** Configuration Debug Rules Permissions

Downlinks + Add Downlink

Enter name、 description、 fields used and payload encoder respectively.

Name: Set User-Defined Sending Time Interval

Description: Set the user-defined report transmission interval and store it in the configuration variable.(5Min-1440Min)

Payload Encoder: copy in [Github](#).

Configure Downlink

Name
Set User-Defined Sending Time Interval

Description
Set the user-defined report transmission interval and store it in the configuration variable. (5Min-1440Min)

Fields used
If your encoder function takes input from the device's fields, you can specify them here. They will be used to create the form for the downlink generator.
SENDING_TIME_INTERVAL

☐ Trigger on measurements

Port
1

Payload Encoder

```

1 function Encoder(measurements, port) {
2   var interval = measurements["SENDING_TIME_INTERVAL"].value * 60;
3   if (interval < 300) {
4     interval = 300;
5     console.log("Interval < 300 Seconds / 5 Minutes not allowed!");
6   }
7   // Convert to hexadecimal only from interval
8   return interval.toString(16).padStart(4, '0').match(/.{2}/g).map(function(f) { return parseInt(f, 16);
9
10  /**
11   * String.prototype.padStart() polyfill
12   * https://github.com/uxitten/polyfill/blob/master/string.polyfill.js
13   * https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String/padStart
14   */
15   if (!String.prototype.padStart) {
16     String.prototype.padStart = function padStart(targetLength,padString) {
17       targetLength = targetLength>=0 ? targetLength : 0;
18       padString = String(typeof padString !== 'undefined' ? padString : ' ');
19       if (this.length > targetLength) {
20         return String(this);
21       }
22       else {
23         targetLength = targetLength-this.length;
24         if (targetLength > padString.length) {
25           padString = padString.repeat(targetLength/padString.length); //append to original to end
26         }

```

3、Click “Dashboard-->switch-->+ Add Widget”.

Select “Downlink” and setting as follow image.

Edit Downlink Widget

User-Defined Time Interval(5Min-1440Min)

Basics | Data | Appearance

Title
English User-Defined Time Interval(5Min-1440Min)
German
+ Add Translation

Cancel Save

Edit Downlink Widget

User-Defined Time Interval(5Min-1440Min)

Basics | **Data** | Appearance

Downlink
Set User-Defined Sending Time Interval

Additional Downlinks
+ Add

Cancel Save

4、Click the switch to save, and you can click to change your time Interval.

User-Defined Time Interval(5Min-1440Min)

Sending Time Interval

1

Cancel Save measurements and send downlink