



RS485-BL – Waterproof RS485 to LoRaWAN Converter

last modified by Xiaoling

on 2023/02/21 15:12

Table of Contents

1. Introduction	5
1.1 What is RS485-BL RS485 to LoRaWAN Converter	5
1.2 Specifications	6
1.3 Features	7
1.4 Applications	7
1.5 Firmware Change log	7
1.6 Hardware Change log	7
2. Pin mapping and Power ON Device	7
3. Operation Mode	8
3.1 How it works?	8
3.2 Example to join LoRaWAN network	8
3.3 Configure Commands to read data	15
3.3.1 Configure UART settings for RS485 or TTL communication (Since v1.3.3)	15
3.3.2 Configure sensors	17
3.3.3 Configure read commands for each sampling	18
3.3.4 Compose the uplink payload	22
3.3.5 Uplink on demand	26
3.3.6 Uplink on Interrupt	26
3.4 Uplink Payload	27
3.5 Configure RS485-BL via AT or Downlink	28
3.5.1 Common Commands:	28
3.5.2 Sensor related commands:	28
3.6 Buttons	36
3.7 +3V3 Output (Since v1.3.3)	36
3.8 +5V Output (Since v1.3.3)	36
3.9 LEDs	36
3.10 Switch Jumper	36
4. Case Study	37
5. Use AT Command	37
5.1 Access AT Command	37
5.2 Common AT Command Sequence	38
5.2.1 Multi-channel ABP mode (Use with SX1301/LG308)	38
5.2.2 Single-channel ABP mode (Use with LG01/LG02)	38
6. FAQ	39
6.1 How to upgrade the image?	39
6.2 How to change the LoRa Frequency Bands/Region?	42
6.3 How many RS485-Slave can RS485-BL connects?	42
6.4 How to Use RS485-BL to connect to RS232 devices?	42
6.5 How to judge whether there is a problem with the set COMMAND	42
6.5.1 Introduce:	42
6.5.2 Set up PC to monitor RS485 network With Serial tool	43
6.5.3 With ModRsim2:	44
6.5.4 Example – Test the CFGDEV command	47
6.5.5 Example – Test CMD command sets.	48
6.5.6 Test with PC	49
6.6 Where to get the decoder for RS485-BL?	51
7. Trouble Shooting	51
7.1 Downlink doesn't work, how to solve it?	51
7.2 Why I can't join TTN V3 in US915 /AU915 bands?	51
8. Order Info	51
9. Packing Info	52
10. Support	52



Table of Contents:

- [1. Introduction](#)
 - [1.1 What is RS485-BL RS485 to LoRaWAN Converter](#)
 - [1.2 Specifications](#)
 - [1.3 Features](#)
 - [1.4 Applications](#)
 - [1.5 Firmware Change log](#)
 - [1.6 Hardware Change log](#)
- [2. Pin mapping and Power ON Device](#)

- [3. Operation Mode](#)
 - [3.1 How it works?](#)
 - [3.2 Example to join LoRaWAN network](#)
 - [3.3 Configure Commands to read data](#)
 - [3.3.1 Configure UART settings for RS485 or TTL communication \(Since v1.3.3\)](#)
 - [3.3.2 Configure sensors](#)
 - [3.3.3 Configure read commands for each sampling](#)
 - [3.3.4 Compose the uplink payload](#)
 - [3.3.5 Uplink on demand](#)
 - [3.3.6 Uplink on Interrupt](#)
 - [3.4 Uplink Payload](#)
 - [3.5 Configure RS485-BL via AT or Downlink](#)
 - [3.5.1 Common Commands:](#)
 - [3.5.2 Sensor related commands:](#)
 - [Choose Device Type \(RS485 or TTL\) \(Since v1.3.3\)](#)
 - [RS485 Debug Command \(AT+CFGDEV\)](#)
 - [Set Payload version](#)
 - [Set RS485 Sampling Commands](#)
 - [Fast command to handle MODBUS device](#)
 - [RS485 command timeout](#)
 - [Uplink payload mode](#)
 - [Manually trigger an Uplink](#)
 - [Clear RS485 Command](#)
 - [Set Serial Communication Parameters](#)
 - [Configure Databit\(Since version 1.4.0\)](#)
 - [Encrypted payload](#)
 - [Get sensor value](#)
 - [Resets the downlink packet count](#)
 - [When the limit bytes are exceeded, upload in batches](#)
 - [Copy downlink to uplink](#)
 - [Query version number and frequency band \ TDC](#)
 - [Control output power duration](#)
 - [3.6 Buttons](#)
 - [3.7 +3V3 Output \(Since v1.3.3\)](#)
 - [3.8 +5V Output \(Since v1.3.3\)](#)
 - [3.9 LEDs](#)
 - [3.10 Switch Jumper](#)
- [4. Case Study](#)
- [5. Use AT Command](#)
 - [5.1 Access AT Command](#)
 - [5.2 Common AT Command Sequence](#)
 - [5.2.1 Multi-channel ABP mode \(Use with SX1301/LG308\)](#)
 - [5.2.2 Single-channel ABP mode \(Use with LG01/LG02\)](#)
- [6. FAQ](#)
 - [6.1 How to upgrade the image?](#)
 - [6.2 How to change the LoRa Frequency Bands/Region?](#)
 - [6.3 How many RS485-Slave can RS485-BL connects?](#)
 - [6.4 How to Use RS485-BL to connect to RS232 devices?](#)
 - [6.5 How to judge whether there is a problem with the set COMMAND](#)
 - [6.5.1 Introduce:](#)
 - [6.5.2 Set up PC to monitor RS485 network With Serial tool](#)
 - [6.5.3 With ModRSim2:](#)
 - [6.5.4 Example – Test the CFGDEV command](#)
 - [6.5.5 Example – Test CMD command sets.](#)
 - [6.5.6 Test with PC](#)
 - [6.6 Where to get the decoder for RS485-BL?](#)
- [7. Trouble Shooting](#)
 - [7.1 Downlink doesn't work, how to solve it?](#)
 - [7.2 Why I can't join TTN V3 in US915 /AU915 bands?](#)
- [8. Order Info](#)

- [9. Packing Info](#)
- [10. Support](#)

1. Introduction

1.1 What is RS485-BL RS485 to LoRaWAN Converter

The Dragino RS485-BL is a **RS485 / UART to LoRaWAN Converter** for Internet of Things solutions. User can connect RS485 or UART sensor to RS485-BL converter, and configure RS485-BL to periodically read sensor data and upload via LoRaWAN network to IoT server.

RS485-BL can interface to RS485 sensor, 3.3v/5v UART sensor or interrupt sensor. RS485-BL provides **a 3.3v output** and **a 5v output** to power external sensors. Both output voltages are controllable to minimize the total system power consumption.

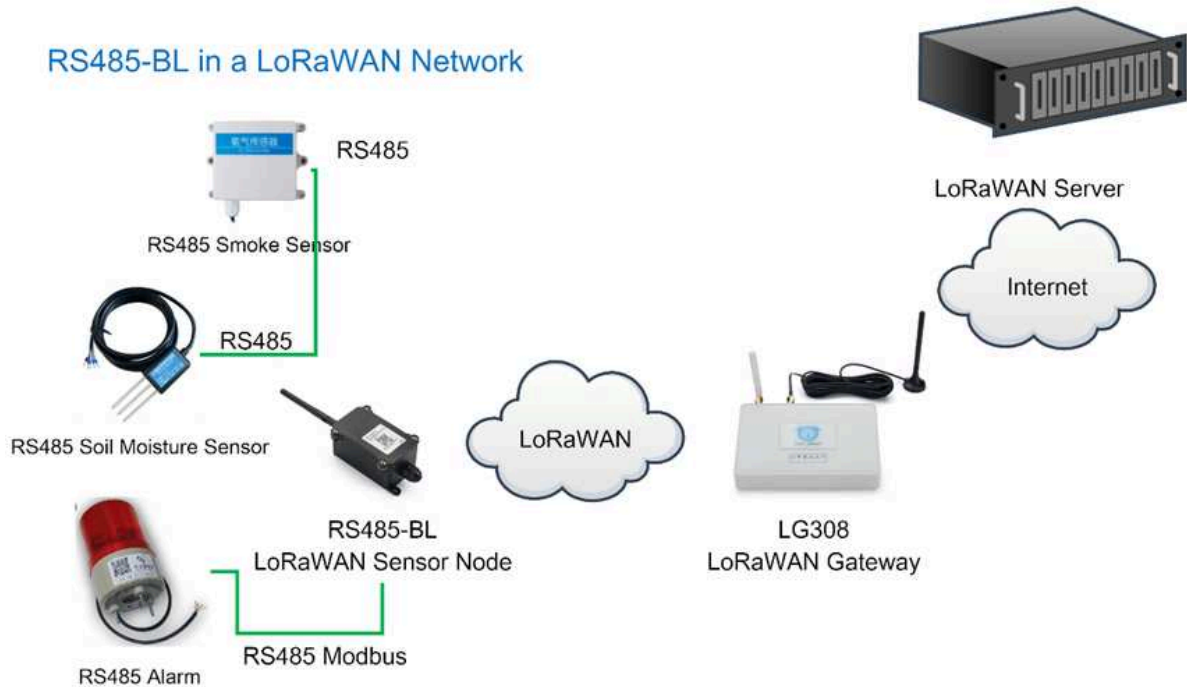
RS485-BL is IP67 **waterproof** and powered by **8500mAh Li-SOCI2 battery**, it is designed for long term use for several years.

RS485-BL runs standard **LoRaWAN 1.0.3 in Class A**. It can reach long transfer range and easy to integrate with LoRaWAN compatible gateway and IoT server.

For data uplink, RS485-BL sends user-defined commands to RS485 devices and gets the return from the RS485 devices. RS485-BL will process these returns data according to user-define rules to get the final payload and upload to LoRaWAN server.

For data downlink, RS485-BL runs in LoRaWAN Class A. When there is downlink commands from LoRaWAN server, RS485-BL will forward the commands from LoRaWAN server to RS485 devices.

Each RS485-BL pre-load with a set of unique keys for LoRaWAN registration, register these keys to LoRaWAN server and it will auto connect after power on.



1.2 Specifications

Hardware System:

- STM32L072xxxx MCU
- SX1276/78 Wireless Chip
- Power Consumption (exclude RS485 device):
 - Idle: 6uA@3.3v
 - 20dB Transmit: 130mA@3.3v
- 5V sampling maximum current: 500mA

Interface for Model:

- 1 x RS485 Interface
- 1 x TTL Serial , 3.3v or 5v.
- 1 x I2C Interface, 3.3v or 5v.
- 1 x one wire interface
- 1 x Interrupt Interface
- 1 x Controllable 5V output, max

LoRa Spec:

- Frequency Range:
 - Band 1 (HF): 862 ~ 1020 Mhz
 - Band 2 (LF): 410 ~ 528 Mhz
- 168 dB maximum link budget.
- +20 dBm - 100 mW constant RF output vs.
- Programmable bit rate up to 300 kbps.
- High sensitivity: down to -148 dBm.

- Bullet-proof front end: IIP3 = -12.5 dBm.
- Excellent blocking immunity.
- Fully integrated synthesizer with a resolution of 61 Hz.
- LoRa modulation.
- Built-in bit synchronizer for clock recovery.
- Preamble detection.
- 127 dB Dynamic Range RSSI.
- Automatic RF Sense and CAD with ultra-fast AFC.

1.3 Features

- LoRaWAN Class A & Class C protocol (default Class A)
- Frequency Bands: CN470/EU433/KR920/US915/EU868/AS923/AU915/IN865/RU864/MA869
- AT Commands to change parameters
- Remote configure parameters via LoRaWAN Downlink
- Firmware upgradable via program port
- Support multiply RS485 devices by flexible rules
- Support Modbus protocol
- Support Interrupt uplink

1.4 Applications

- Smart Buildings & Home Automation
- Logistics and Supply Chain Management
- Smart Metering
- Smart Agriculture
- Smart Cities
- Smart Factory

1.5 Firmware Change log

[RS485-BL Image files – Download link and Change log](#)

1.6 Hardware Change log

v1.4

1. Change Power IC to TPS22916

v1.3

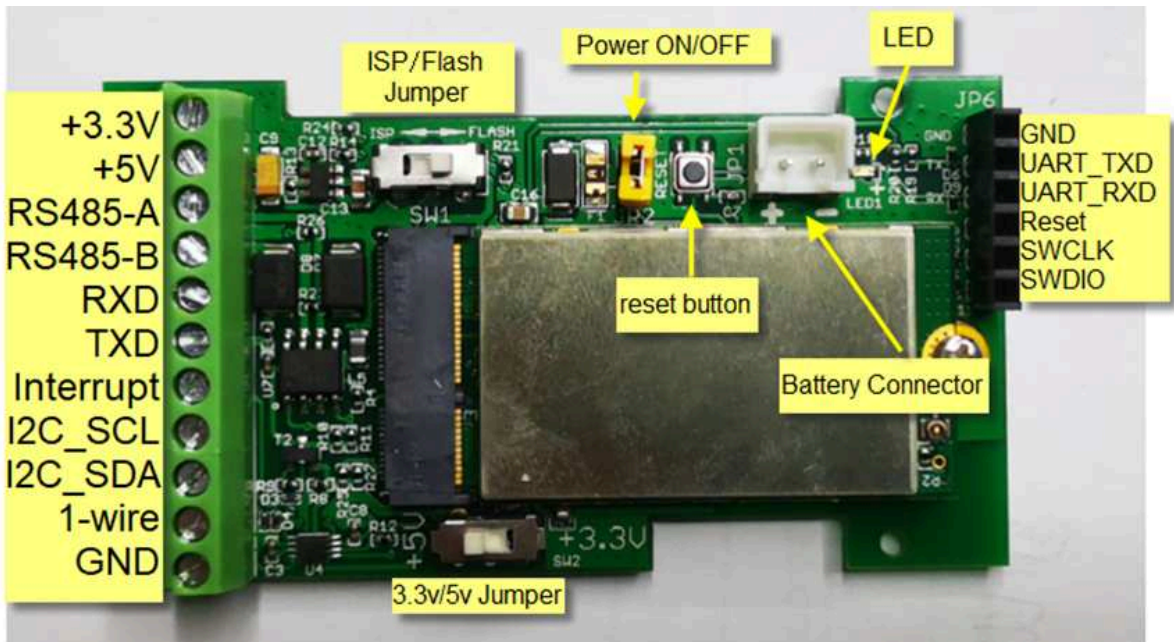
1. Change JP3 from KF350-8P to KF350-11P, Add one extra interface for I2C and one extra interface for one-wire

v1.2

Release version

2. Pin mapping and Power ON Device

The RS485-BL is powered on by 8500mAh battery. To save battery life, RS485-BL is shipped with power off. User can put the jumper to power on RS485-BL.



The Left TXD and RXD are TTL interface for external sensor. TTL level is controlled by 3.3/5v Jumper.

3. Operation Mode

3.1 How it works?

The RS485-BL is configured as LoRaWAN OTAA Class A mode by default. It has OTAA keys to join network. To connect a local LoRaWAN network, user just need to input the OTAA keys in the network server and power on the RS485-BL. It will auto join the network via OTAA.

3.2 Example to join LoRaWAN network

Here shows an example for how to join the TTN V3 Network. Below is the network structure, we use [LG308](#) as LoRaWAN gateway here.



The RS485-BL in this example connected to two RS485 devices for demonstration, user can connect to other RS485 devices via the same method.

The LG308 is already set to connect to [TTN V3 network](#). So what we need to now is only configure the TTN V3:

Step 1: Create a device in TTN V3 with the OTAA keys from RS485-BL.

Each RS485-BL is shipped with a sticker with unique device EUI:



User can enter this key in their LoRaWAN Server portal. Below is TTN V3 screen shot:

Add APP EUI in the application.

S
K

THE THINGS STACK
Community Edition

Overview Applications Gateways Orga

Add application

Owner*

davidhuang

Application ID*

my-new-application

Application name


My new application

Description

Description for my new application

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

Create application

 **CCC**
ID: 123

4 End devices 2 Collaborators 2 API keys Created 95 days ago

General information

Application ID	<input type="text" value="123"/>
Created at	Feb 2, 2021 11:12:30
Last updated at	Apr 30, 2021 11:00:33

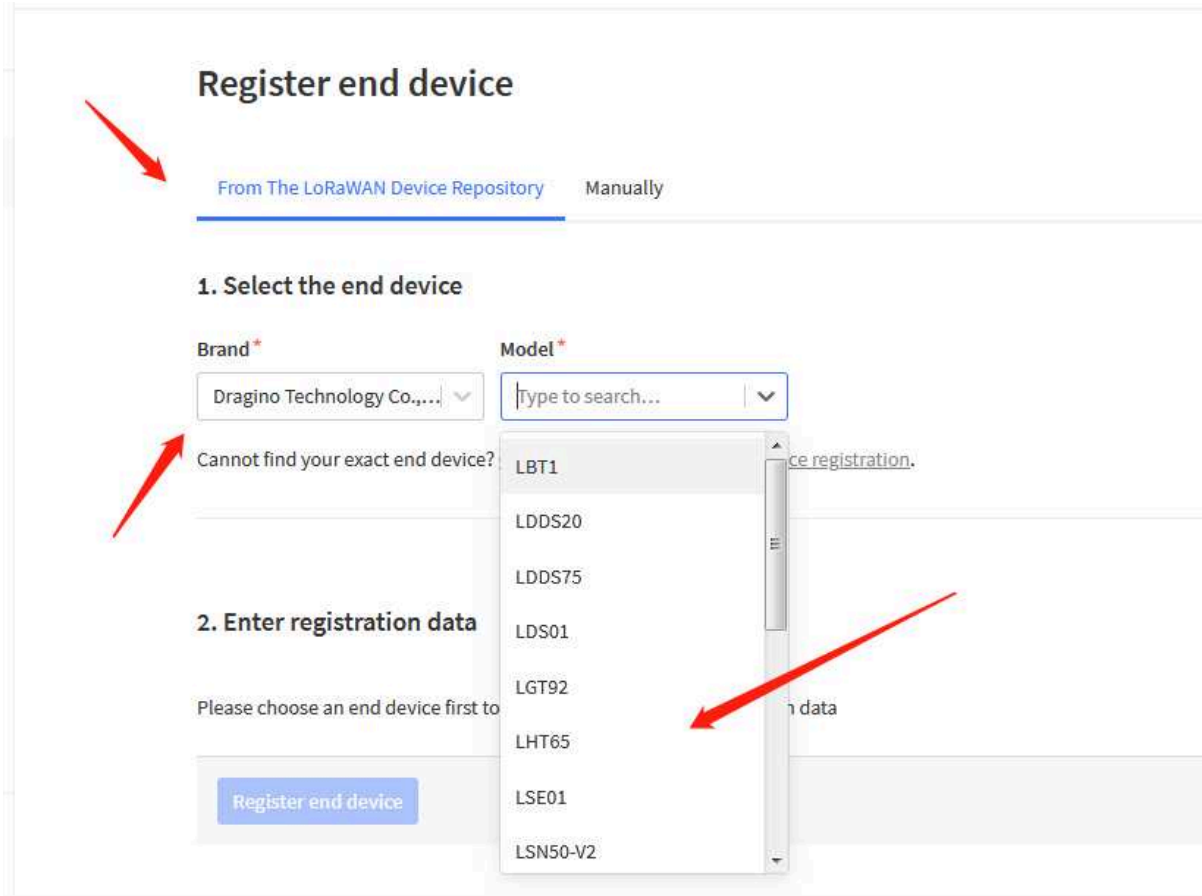
Live data See all activity →

- ↑ 10:09:42 1231234234... Forward data message to Application Server
- ⓘ 10:09:42 1231234234... Store upstream data message
- ↑ 10:09:42 1231234234... Forward uplink data message
- ↑ 10:09:42 1231234234... Receive uplink data message
- ↑ 10:09:42 1231234234... Successfully processed data message
- ↑ 10:09:42 1231234234... Drop data message

End devices (4)

Search by ID Import end devices **Add end device**

ID ↕	Name ↕	DevEUI	JoinEUI	Created ↕
------	--------	--------	---------	-----------



You can also choose to create the device manually.

Register end device

From The LoRaWAN Device Repository [Manually](#)

Preparation

Activation mode *

- Over the air activation (OTAA)
- Activation by personalization (ABP)
- Multicast
- Do not configure activation

LoRaWAN version ⓘ *

Network Server address

Application Server address

External Join Server ⓘ

Add APP KEY and DEV EUI

2. Enter registration data

Frequency plan  *

Europe 863-870 MHz (SF12 for RX2) 

The frequency plan used by the end device

AppEUI  *

.. 00

The AppEUI uniquely identifies the owner of the end device. If no AppEUI is provided by the device manufacturer (usually for dev

DevEUI  *

..

The DevEUI is the unique identifier for this end device

AppKey  *

.. 

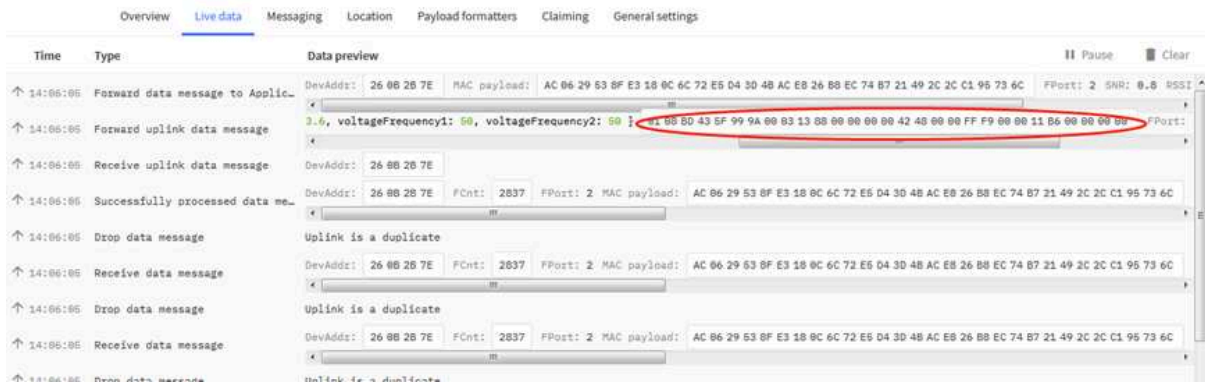
The root key to derive session keys to secure communication between the end device and the application

End device ID *

my-new-device

After registration

Step 2: Power on RS485-BL and it will auto join to the TTN V3 network. After join success, it will start to upload message to TTN V3 and user can see in the panel.



The screenshot shows the 'Live data' tab in the TTN V3 interface. The 'Data preview' section displays a list of messages with their details. A red circle highlights the MAC payload of a message, which is: AC 06 29 53 8F E3 18 0C 6C 72 E5 D4 3D 4B AC E8 26 88 EC 74 B7 21 49 2C 2C C1 95 73 6C. The interface also shows other messages like 'Forward data message to Applic...', 'Forward uplink data message', 'Receive uplink data message', 'Successfully processed data me...', 'Drop data message', and 'Receive data message'.

3.3 Configure Commands to read data

There are plenty of RS485 and TTL level devices in the market and each device has different command to read the valid data. To support these devices in flexible, RS485-BL supports flexible command set. User can use [AT Commands or LoRaWAN Downlink](#) Command to configure how RS485-BL should read the sensor and how to handle the return from RS485 or TTL sensors.

3.3.1 Configure UART settings for RS485 or TTL communication (Since v1.3.3)

RS485-BL can connect to either RS485 sensors or TTL sensor. User need to specify what type of sensor need to connect.

1. RS485-MODBUS mode:

AT+MOD=1 // Support RS485-MODBUS type sensors. User can connect multiply RS485 , Modbus sensors to the A / B pins.

2. TTL mode:

AT+MOD=2 // Support TTL Level sensors, User can connect one TTL Sensor to the TXD/RXD/GND pins.

RS485-BL default UART settings is **9600, no parity, stop bit 1**. If the sensor has a different settings, user can change the RS485-BL setting to match.

AT Commands	Description	Example
AT+BAUDR	Set the baud rate (for RS485 connection). Default Value is: 9600.	AT+BAUDR=9600 Options: (1200,2400,4800,14400,19200,115200)
AT+PARITY	Set UART parity (for RS485 connection) Default Value is: no parity.	AT+PARITY=0 Option: 0: no parity, 1: odd parity, 2: even parity
AT+STOPBIT	Set serial stopbit (for RS485 connection) Default Value is: 1bit.	AT+STOPBIT=0 for 1bit AT+STOPBIT=1 for 1.5 bit AT+STOPBIT=2 for 2 bits

Example (Soil three-parameter detector) :

Wiring the UART sensor

```

GND <-----> GND
TX  <-----> RX
RX  <-----> TX
VCC <-----> 3.3/5V
    
```



Set the correct configuration:

AT+BAUDR=9600

AT+PARITY=0

AT+STOPBIT=1

If the sensor needs 5v. Need to move the switch position to 5v and then use the command

AT+5VT=30000

Configuration read command:

AT+CFGDEV=FE 03 00 00 00 03 11 C4,0

FE: Station address

03: Function code

00 00: Register start address

00 03: Number of registers

11 04: Check code

```
AT+CFGDEV=FE 03 00 00 00 03 11 C4,0
AT+CFGDEV=fe 03 00 00 00 03 11 c4 ,0
RETURN DATA:
fe 03 06 00 00 09 49 00 00 b6 cb
OK
```


Use AT+COMMAND1 to set it as a command, and use AT+DATACUT1 to intercept the bytes I need

```

AT+SETMAXNBTRANS=1,0
AT+DISFCNTCHECK=0
AT+DISMACANS=0
AT+COMMAND1=fe 03 00 00 00 03 11 c4 ,0      AT+SEARCH1=0,0
AT+DATACUT1=11,2,4~9      AT+CMDDL1=0
AT+COMMAND2=0,0      AT+SEARCH2=0,0      AT+DATACUT2=0,0,0      AT+CMDDL2=0
AT+COMMAND3=0,0      AT+SEARCH3=0,0      AT+DATACUT3=0,0,0      AT+CMDDL3=0
AT+COMMAND4=0,0      AT+SEARCH4=0,0      AT+DATACUT4=0,0,0      AT+CMDDL4=0
AT+COMMAND5=0,0      AT+SEARCH5=0,0      AT+DATACUT5=0,0,0      AT+CMDDL5=0
AT+COMMAND6=0,0      AT+SEARCH6=0,0      AT+DATACUT6=0,0,0      AT+CMDDL6=0
AT+COMMAND7=0,0      AT+SEARCH7=0,0      AT+DATACUT7=0,0,0      AT+CMDDL7=0
AT+COMMAND8=0,0      AT+SEARCH8=0,0      AT+DATACUT8=0,0,0      AT+CMDDL8=0
AT+COMMAND9=0,0      AT+SEARCH9=0,0      AT+DATACUT9=0,0,0      AT+CMDDL9=0
AT+COMMANDA=0,0      AT+SEARCHA=0,0      AT+DATACUTA=0,0,0      AT+CMDDLA=0
AT+COMMANDB=0,0      AT+SEARCHB=0,0      AT+DATACUTB=0,0,0      AT+CMDDLb=0
AT+COMMANDC=0,0      AT+SEARCHC=0,0      AT+DATACUTC=0,0,0      AT+CMDDLC=0
    
```

upload payload:

```

[1870882]***** UpLinkCounter= 31 *****
[1870884]TX on freq 904900000 Hz at DR 3
[1870945]txDone
[1875932]RX on freq 926300000 Hz at DR 13
[1875957]rxTimeOut
[1876948]RX on freq 923300000 Hz at DR 8
[1877008]rxTimeOut

CMD1      = fe 03 00 00 00 03 11 c4
RETURN1   = fe 03 06 00 00 09 49 00 00 b6 cb
Payload   = 0d 32 01 00 00 09 49 00 00
    
```

3.3.2 Configure sensors

Some sensors might need to configure before normal operation. User can configure such sensor via PC or through RS485-BL AT Commands [AT+CFGDEV](#).

When user issue an [AT+CFGDEV](#) command, Each [AT+CFGDEV](#) equals to send a command to the RS485 or TTL sensors. This command will only run when user input it and won't run during each sampling.

AT Commands	Description	Example
AT+CFGDEV	This command is used to configure the RS485/TTL devices; they won't be used during sampling.	AT+CFGDEV=xx xx xx xx xx xx xx xx xx xx,m

```
AT+CFGDEV=xx xx xx xx xx xx xx xx xx xx,  
mm: 0: no CRC, 1: add CRC-16/MODBUS in the end of  
this command
```

Detail of AT+CFGDEV command see [AT+CFGDEV detail](#).

3.3.3 Configure read commands for each sampling

RS485-BL is a battery powered device; it will sleep most of time. And wake up on each period and read RS485 / TTL sensor data and uplink.

During each sampling, we need to confirm what commands we need to send to the sensors to read data. After the RS485/TTL sensors send back the value, it normally includes some bytes and we only need a few from them for a shorten payload.

To save the LoRaWAN network bandwidth, we might need to read data from different sensors and combine their valid value into a short payload.

This section describes how to achieve above goals.

During each sampling, the RS485-BL can support 15 commands to read sensors. And combine the return to one or several uplink payloads.

Command from RS485-BL to Sensor:

RS485-BL can send out pre-set max 15 strings via **AT+COMMAD1**, **ATCOMMAND2**,..., to **AT+COMMANDF** . All commands are of same grammar.

Handle return from sensors to RS485-BL:

After RS485-BL send out a string to sensor, RS485-BL will wait for the return from RS485 or TTL sensor. And user can specify how to handle the return, by **AT+DATACUT** or **AT+SEARCH** commands

- **AT+DATACUT**

When the return value from sensor have fix length and we know which position the valid value we should get, we can use AT+DATACUT command.

- **AT+SEARCH**

When the return value from sensor is dynamic length and we are not sure which bytes the valid data is, instead, we know what value the valid value following. We can use AT+SEARCH to search the valid value in the return string.

Define wait timeout:

Some RS485 device might has longer delay on reply, so user can use AT+CMDDL to set the timeout for getting reply after the RS485 command is sent. For example, AT+CMDDL1=1000 to send the open time to 1000ms

After we got the valid value from each RS485 commands, we need to combine them together with the command **AT+DATAUP**.

Examples:

Below are examples for the how above AT Commands works.

AT+COMMANDx : This command will be sent to RS485/TTL devices during each sampling, Max command length is 14 bytes. The grammar is:

```
AT+COMMANDx=xx xx xx xx xx xx xx xx xx xx xx,m
xx xx xx xx xx xx xx xx xx xx xx: The RS485 command to be sent
m: 0: no CRC, 1: add CRC-16/MODBUS in the end of this command
```

For example, if we have a RS485 sensor. The command to get sensor value is: 01 03 0B B8 00 02 46 0A. Where 01 03 0B B8 00 02 is the Modbus command to read the register 0B B8 where stored the sensor value. The 46 0A is the CRC-16/MODBUS which calculate manually.

In the RS485-BL, we should use this command AT+COMMAND1=01 03 0B B8 00 02,1 for the same.

AT+SEARCHx: This command defines how to handle the return from AT+COMMANDx.

```
AT+SEARCHx=aa,xx xx xx xx xx
• aa: 1: prefix match mode; 2: prefix and suffix match mode
• xx xx xx xx xx: match string. Max 5 bytes for prefix and 5 bytes for suffix
```

Examples:

1) For a return string from AT+COMMAND1: 16 0c 1e 56 34 2e 30 58 5f 36 41 30 31 00 49

If we set AT+SEARCH1=1,1E 56 34. (max 5 bytes for prefix)

The valid data will be all bytes after 1E 56 34 , so it is **2e 30 58 5f 36 41 30 31 00 49**

```
CMD1    = 11 01 1e d0
SEARCH1 = 1e 56 34
RETURN1 = 2e 30 58 5f 36 41 30 31 00 49
Payload = 8d 2d 01 2e 30 58 5f 36 41 30 31 00 49
```

2) For a return string from AT+COMMAND1: 16 0c 1e 56 34 2e 30 58 5f 36 41 30 31 00 49

If we set AT+SEARCH1=2, 1E 56 34+31 00 49

Device will search the bytes between 1E 56 34 and 31 00 49. So it is **2e 30 58 5f 36 41 30**

```
CMD1    = 11 01 1e d0
SEARCH1 = 1e 56 34 + 31 00 49
RETURN1 = 2e 30 58 5f 36 41 30
Payload = 8d 2f 01 2e 30 58 5f 36 41 30
```

AT+DATA CUTx : This command defines how to handle the return from AT+COMMANDx, max return length is 100 bytes. (Since 1.4.0)

AT+DATA CUTx=a,b,c

- a: length for the return of AT+COMMAND
- b:1: grab valid value by byte, max 6 bytes. 2: grab valid value by bytes section, max 3 sections.
- c: define the position for valid value.

Examples:

- **Grab bytes:**

```

AT+PAYVER=1
AT+COMMAND1=01 03 0b b8 00 02 ,1 AT+DATA CUT1=10,1,9+4+6+8+1+3
AT+COMMAND2=0,0 AT+DATA CUT2=0,0,0
AT+COMMAND3=0,0 AT+DATA CUT3=0,0,0
AT+COMMAND4=0,0 AT+DATA CUT4=0,0,0
AT+COMMAND5=0,0 AT+DATA CUT5=0,0,0
/ AT+DATA CUT1=10,1,9+4+6+8+1+3
/ a=10, return total 10 bytes (20 20 20 20 2d 30 2e 32 20 75)
/ b=1 grab byte.
/ c=9+4+6+8+1+3 (grab the 9th , 4th, 6th, 8th, 1th, 3rd byte and link them together by grab sequence
/ so command1 valid value is 20 20 30 32 20 20
AT+COMMANDE=0,0 AT+DATA CUT6=0,0,0
AT+COMMANDE=0,0 AT+DATA CUT6=0,0,0
AT+COMMANDE=0,0 AT+DATA CUT6=0,0,0
AT+CHS=0
OK
CMD1 = 01 03 0b b8 00 02 46 0a
RETURN1 = 20 20 20 20 2d 30 2e 32 20 75
Payload = 0c fc 01 20 20 30 32 20 20
    
```

- **Grab a section.**

```

AT+PAYVER=1
AT+COMMAND1=01 03 0b b8 00 02 ,1 AT+DATA CUT1=8,2,4~8
AT+COMMAND2=0,0 AT+DATA CUT2=0,0,0
AT+COMMAND3=0,0 AT+DATA CUT3=0,0,0
AT+COMMAND4=0,0 AT+DATA CUT4=0,0,0
AT+COMMAND5=0,0 AT+DATA CUT5=0,0,0
AT+COMMAND6=0,0 AT+DATA CUT6=0,0,0
AT+COMMAND7=0,0 AT+DATA CUT7=0,0,0
AT+COMMAND8=0,0 AT+DATA CUT8=0,0,0
AT+CC AT+DATA CUT1=8,2,4~8
AT+CC
AT+CC a=8, return total 8 bytes (20 20 20 20 2d 30 2e 00)
AT+CC b=2
AT+CC
AT+CC c=4~8 (grap the 4th ~ 8th bytes from return, so command1 valid value is 20 2d 30 2e 00)
AT+COMMANDF=0,0 AT+DATA CUTF=0,0,0
AT+CHS=0

```

OK

```

CMD1      = 01 03 0b b8 00 02 46 0a
RETURN1   = 20 20 20 20 2d 30 2e 00
Payload   = 0c fc 01 20 2d 30 2e 00

```

- Grab different sections.

```

AT+COMMAND1=01 03 0b b8 00 02 ,1 AT+DATA CUT1=13,2,1~2+4~7+10~11
AT+COMMAND2=0,0 AT+DATA CUT2=0,0,0
AT+COMMAND3=0,0 AT+DATA CUT3=0,0,0
AT+COMMAND4=0,0 AT+DATA CUT4=0,0,0
AT+COMMAND5=0,0 AT+DATA CUT5=0,0,0
AT+COMMAND6=0,0 AT+DATA CUT6=0,0,0
AT+DATA CUT1=13,2,1~2+4~7+10~11
a=13, return total 13 bytes (90 02 6a 82 1a 04 20 2d 30 2e dd 9b 00)
b=2
c=1~2+4~7+10~11 (grap the 1 ~ 2 bytes + 4~7 bytes + 10~11 bytes
so command1 valid value is 90 02 82 1a 04 20 2e dd)
AT+COMMANDF=0,0 AT+DATA CUTF=0,0,0
AT+COMMANDF=0,0 AT+DATA CUTF=0,0,0
AT+CHS=0

```

OK

```

CMD1      = 01 03 0b b8 00 02 46 0a
RETURN1   = 90 02 6a 82 1a 04 20 2d 30 2e dd 9b 00
Payload   = 0c fc 01 90 02 82 1a 04 20 2e dd

```

Note:

AT+SEARCHx and **AT+DATA CUTx** can be used together, if both commands are set, RS485-BL will first process **AT+SEARCHx** on the return string and get a temporary string, and then process **AT+DATA CUTx** on this temporary string to get the final payload. In this case, **AT+DATA CUTx** need to set to format **AT+DATA CUTx=0,xx,xx** where the return bytes set to **0**.

Example:

AT+COMMAND1=11 01 1E D0,0

AT+SEARCH1=1,1E 56 34

AT+DATACUT1=0,2,1~5

Return string from AT+COMMAND1: 16 0c 1e 56 34 2e 30 58 5f 36 41 30 31 00 49

String after SEARCH command: 2e 30 58 5f 36 41 30 31 00 49

Valid payload after DataCUT command: 2e 30 58 5f 36

```
CMD1    = 11 01 1e d0
SEARCH1 = 1e 56 34
RETURN1 = 2e 30 58 5f 36 41 30 31 00 49
Payload = 8d 2d 01 2e 30 58 5f 36
```

3.3.4 Compose the uplink payload

Through AT+COMMANDx and AT+DATACUTx we got valid value from each RS485 commands, Assume these valid value are RETURN1, RETURN2, ..., to RETURNx. The next step is how to compose the LoRa Uplink Payload by these RETURNS. The command is **AT+DATAUP**.

Examples: AT+DATAUP=0

Compose the uplink payload with value returns in sequence and send with **A SINGLE UPLINK**.

Final Payload is

Battery Info+PAYVER + VALID Value from RETURN1 + Valid Value from RETURN2 + ... + RETURNx

Where PAYVER is defined by AT+PAYVER, below is an example screen shot.


```

AT+PARITY=0
AT+DATAUP=0
AT+PAYVER=1
AT+COMMAND1=01 03 0b b8 00 02 ,1
AT+COMMAND2=0,0
AT+COMMAND3=0,0
AT+COMMAND4=0,0
AT+COMMAND5=0,0
AT+COMMAND6=0,0
AT+COMMAND7=0,0
AT+COMMAND8=0,0
AT+COMMAND9=0,0
AT+COMMANDA=01 03 0b b8 00 02 ,1
AT+COMMANDB=0,0
AT+COMMANDC=0,0
AT+COMMANDD=0,0
AT+COMMANDE=0,0
AT+COMM=0,0
AT+CHS= Valid value from RETURN1=20 20 30 32 20 20
OK
CMD1 = 01 03 0b b8 00 02 46 0a
RETURN1 = 20 20 20 20 2d 30 2e 32 20 00
CMD10 = 01 03 0b b8 00 02 46 0a
RETURN10 = 20 20 20 20 2d 30 2e 33 20 75 41 20 0d 0a 20 00
Payload = 0c fc 01 20 20 30 32 20 20 20 20 20 20 30 2e 33 20 75 41 20

```

Battery Info
PAYVER
Valid Value from Return1
Valid Value from Return10

Examples: AT+DATAUP=1

Compose the uplink payload with value returns in sequence and send with **Multiply UPLINKS**.

Final Payload is

Battery Info+PAYVER + PAYLOAD COUNT + PAYLOAD# + DATA

1. Battery Info (2 bytes): Battery voltage
2. PAYVER (1 byte): Defined by AT+PAYVER
3. PAYLOAD COUNT (1 byte): Total how many uplinks of this sampling.
4. PAYLOAD# (1 byte): Number of this uplink. (from 0,1,2,3...,to PAYLOAD COUNT)
5. DATA: Valid value: max 6 bytes(US915 version here, Notice*!) for each uplink so each uplink <= 11 bytes. For the last uplink, DATA will might less than 6 bytes

```

AT+DATAUP=1
AT+PAYVER=1
AT+COMMAND1=01 03 0b b8 00 02 ,1 AT+DATAACUT1=10,1,9+4+6+8+1+3
AT+COMMAND2=0,0 AT+DATAACUT2=0,0,0
AT+COMMAND3=0,0 AT+DATAACUT3=0,0,0
AT+COMMAND4=0,0 AT+DATAACUT4=0,0,0
AT+COMMAND5=0,0 AT+DATAACUT5=0,0,0
AT+COMMAND6=0,0 AT+DATAACUT6=0,0,0
AT+COMMAND7=0,0 AT+DATAACUT7=0,0,0
AT+COMMAND8=0,0 AT+DATAACUT8=0,0,0
AT+COMMAND9=0,0 AT+DATAACUT9=0,0,0
AT+COMMANDA=01 03 0b b8 00 02 ,1 AT+DATAACUTA=16,2,1~4+6~12
AT+COMMANDB=0,0 AT+DATAACUTB=0,0,0
AT+COMMANDC=0,0 AT+DATA Valid value from RETURN10=02 aa 05 81 0a 20 20 20 2d 30
AT+COMMANDD=0,0 AT+DATAACUTD=0,0,0
AT Valid value from RETURN1=20 20 0a 33 90 41
AT
AT+CHS=0
OK
CMD1 = 01 03 0b b8 00 02 46 0a
RETURN1 = 90 75 41 20 0d 0a 2e 33 20 00
CMD10 = 01 03 0b b8 00 02 46 0a
RETURN10 = 02 aa 05 81 0d 0a 20 20 20 2d 30 2e 34 20 00
Payload = 0c fc 01 03 00 20 20 0a 33 90 41 02 aa
[2559235]***** upLinkCounter= 85 *****

```

So totally there will be 3 uplinks for this sampling, each uplink includes 6 bytes DATA

DATA1=RETURN1 Valid Value = 20 20 0a 33 90 41

DATA2=1st ~ 6th byte of Valid value of RETURN10= 02 aa 05 81 0a 20

DATA3=7th ~ 11th bytes of Valid value of RETURN10 = 20 20 20 2d 30

Below are the uplink payloads:


```

CMD1 = 01 03 0b b8 00 02 46 0a
RETURN1 = 90 75 41 20 0d 0a 2e 33 20 00
CMD10 = 01 03 0b b8 00 02 46 0a
RETURN10 = 02 aa 05 81 0d 0a 20 20 20 20 2d 30 2e 34 20 00
Payload = 0c fc 01 03 00 20 20 0a 33 90 41
} First Uplink

[2559235]***** UpLinkCounter= 85 *****
[2559257]TX on freq 923400000 Hz at DR 2
[2559666]RX on freq 923200000 Hz at DR 2
[2559668]txDone
[2560671]RX on freq 923400000 Hz at DR 2
[2560732]RX on freq 923200000 Hz at DR 2
[2560734]rxTimeout

CMD1 = 01 03 0b b8 00 02 46 0a
RETURN1 = 90 75 41 20 0d 0a 2e 33 20 00
CMD10 = 01 03 0b b8 00 02 46 0a
RETURN10 = 02 aa 05 81 0d 0a 20 20 20 20 2d 30 2e 34 20 00
Payload = 0c fc 01 03 01 02 aa 05 81 0a 20
} Second Uplink

[2565375]***** UpLinkCounter= 86 *****
[2565395]TX on freq 922800000 Hz at DR 2
[2565803]RX on freq 923200000 Hz at DR 2
[2565806]txDone
[2566809]RX on freq 922800000 Hz at DR 2
[2566870]RX on freq 923200000 Hz at DR 2
[2566872]rxTimeout

CMD1 = 01 03 0b b8 00 02 46 0a
RETURN1 = 90 75 41 20 0d 0a 2e 33 20 00
CMD10 = 01 03 0b b8 00 02 46 0a
RETURN10 = 02 aa 05 81 0d 0a 20 20 20 20 2d 30 2e 34 20 00
Payload = 0c fc 01 03 02 20 20 20 2d 30
} Third Uplink

[2571494]***** UpLinkCounter= 87 *****
[2571510]TX on freq 922600000 Hz at DR 2
[2571874]RX on freq 923200000 Hz at DR 2
[2571876]txDone
[2572879]RX on freq 922600000 Hz at DR 2
[2572940]RX on freq 923200000 Hz at DR 2
[2572942]rxTimeout
    
```

Notice: the Max bytes is according to the max support bytes in different Frequency Bands for lowest SF. As below:

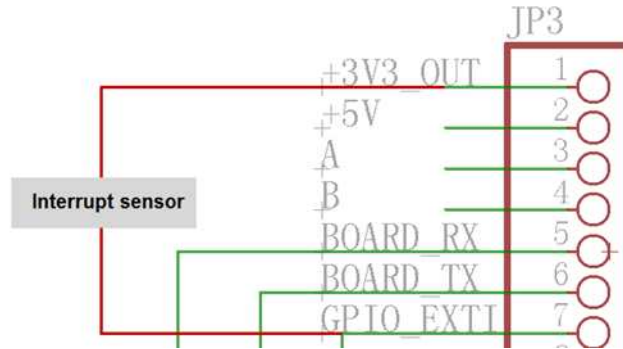
- * For AU915/AS923 bands, if UplinkDwell time=0, max 51 bytes for each uplink (so 51 -5 = 46 max valid date)
- * For AU915/AS923 bands, if UplinkDwell time=1, max 11 bytes for each uplink (so 11 -5 = 6 max valid date).
- * For US915 band, max 11 bytes for each uplink (so 11 -5 = 6 max valid date).
- * For all other bands: max 51 bytes for each uplink (so 51 -5 = 46 max valid date).

*** When AT+DATAUP=1, the maximum number of segments is 15, and the maximum total number of bytes is 1500;**

When AT+DATAUP=1 and AT+ADR=0, the maximum number of bytes of each payload is determined by the DR value. (Since v1.4.0)

- **If the data is empty, return to the display** (Since v1.4.0)

1) When **AT+MOD=1**, if the data intercepted by **AT+DATA CUT** or **AT+MBFUN** is empty, it will display **NULL**, and the payload will be filled with **n FFs**.



AT+INTMOD=0 Disable Interrupt

AT+INTMOD=1 Interrupt trigger by rising or falling edge.

AT+INTMOD=2 Interrupt trigger by falling edge. (Default Value)

AT+INTMOD=3 Interrupt trigger by rising edge.

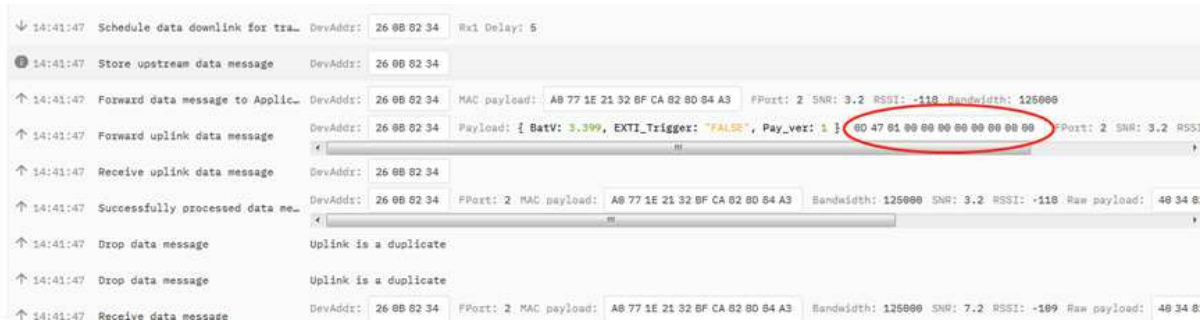
3.4 Uplink Payload

Size(bytes)	2	1	Length depends on the return from the commands
Value	Battery(mV) & Interrupt_Flag	PAYLOAD_VER	If the valid payload is too long and exceed the maximum support payload length in server, server will show payload not provided in the LoRaWAN server.

Below is the decoder for the first 3 bytes. The rest bytes are dynamic depends on different RS485 sensors.

```
function Decoder(bytes, port) {
//Payload Formats of RS485-BL Deceive
return {
//Battery,units:V
BatV:((bytes[0]<<8 | bytes[1])&0x7fff)/1000,
//GPIO_EXTI
EXTI_Trigger:(bytes[0] & 0x80)? "TRUE":"FALSE",
//payload of version
Pay_ver:bytes[2],
};
}
```

TTN V3 uplink screen shot.



3.5 Configure RS485-BL via AT or Downlink

User can configure RS485-BL via AT Commands or LoRaWAN Downlink Commands

There are two kinds of Commands:

- **Common Commands:** They should be available for each sensor, such as: change uplink interval, reset device. For firmware v1.3, user can find what common commands it supports: [AT Commands and Downlink Command](#)
- **Sensor Related Commands:** These commands are special designed for RS485-BL. User can see these commands below:

3.5.1 Common Commands:

They should be available for each of Dragino Sensors, such as: change uplink interval, reset device. For firmware v1.3, user can find what common commands it supports: [End Device AT Commands and Downlink Command](#)

3.5.2 Sensor related commands:

Choose Device Type (RS485 or TTL) (Since v1.3.3)

RS485-BL can connect to either RS485 sensors or TTL sensor. User need to specify what type of sensor need to connect.

- **AT Command**

AT+MOD=1 // Set to support RS485-MODBUS type sensors. User can connect multiply RS485 , Modbus sensors to the A / B pins.

AT+MOD=2 // Set to support TTL Level sensors, User can connect one TTL Sensor to the TXD/RXD/GND pins.

- **Downlink Payload**

0A aa --> same as AT+MOD=aa

RS485 Debug Command (AT+CFGDEV)

This command is used to configure the RS485 or TTL sensors; they won't be used during sampling. Max Length of AT+CFGDEV is **40 bytes**.

- **AT Command**

AT+CFGDEV=xx xx xx xx xx xx xx xx xx xx xx xx,m m: 0: no CRC; 1: add CRC-16/MODBUS in the end of this command.

- **Downlink Payload**

Format: **A8 MM NN XX XX XX XX YY**

Where:

- MM: 1: add CRC-16/MODBUS ; 0: no CRC
- NN: The length of RS485 command
- XX XX XX XX: RS485 command total NN bytes
- YY: How many bytes will be uplink from the return of this RS485 command, if YY=0, RS485-BL will execute the downlink command without uplink; if YY>0, RS485-BL will uplink total YY bytes from the output of this RS485 command

Example 1:

To connect a Modbus Alarm with below commands.

- The command to active alarm is: 0A 05 00 04 00 01 **4C B0**. Where 0A 05 00 04 00 01 is the Modbus command to read the register 00 40 where stored the DI status. The 4C B0 is the CRC-16/MODBUS which calculate manually.
- The command to deactivate alarm is: 0A 05 00 04 00 00 **8D 70**. Where 0A 05 00 04 00 00 is the Modbus command to read the register 00 40 where stored the DI status. The 8D 70 is the CRC-16/MODBUS which calculate manually.

So if user want to use downlink command to control to RS485 Alarm, he can use:

A8 01 06 0A 05 00 04 00 01 00: to activate the RS485 Alarm

A8 01 06 0A 05 00 04 00 00 00: to deactivate the RS485 Alarm

A8 is type code and 01 means add CRC-16/MODBUS at the end, the 3rd byte is 06, means the next 6 bytes are the command to be sent to the RS485 network, the final byte 00 means this command don't need to acquire output.

Example 2:

Check TTL Sensor return:

```
AT+CFGDEV=11 01 1e d0 ,0
RETURN DATA:16 0c 1e 56 34 2e 30 58 5f 36 41 30 31 00 49
```

OK

Set Payload version

This is the first byte of the uplink payload. RS485-BL can connect to different sensors. User can set the PAYVER field to tell server how to decode the current payload.

- **AT Command:**

AT+PAYVER: Set PAYVER field = 1

- **Downlink Payload:**

0xAE 01 --> Set PAYVER field = 0x01

0xAE 0F --> Set PAYVER field = 0x0F

Set RS485 Sampling Commands

AT+COMMANDx, AT+DATACUTx and AT+SEARCHx

These three commands are used to configure how the RS485-BL polling data from Modbus device. Detail of usage please see : [polling RS485 device](#).

- **AT Command:**

AT+COMMANDx: Configure RS485 read command to sensor.

AT+DATACUTx: Configure how to handle return from RS485 devices.

AT+SEARCHx: Configure search command

- **Downlink Payload:**

0xAF downlink command can be used to set AT+COMMANDx or AT+DATACUTx.

Note : if user use AT+COMMANDx to add a new command, he also need to send AT+DATACUTx downlink.

Format: AF MM NN LL XX XX XX XX YY

Where:

- MM: the ATCOMMAND or AT+DATACUT to be set. Value from 01 ~ 0F,
- NN: 0: no CRC; 1: add CRC-16/MODBUS ; 2: set the AT+DATACUT value.
- LL: The length of AT+COMMAND or AT+DATACUT command
- XX XX XX XX: AT+COMMAND or AT+DATACUT command
- YY: If YY=0, RS485-BL will execute the downlink command without uplink; if YY=1, RS485-BL will execute an uplink after got this command.

Example:

AF 03 01 06 0A 05 00 04 00 01 00: Same as AT+COMMAND3=0A 05 00 04 00 01,1

AF 03 02 06 10 01 05 06 09 0A 00: Same as AT+DATACUT3=16,1,5+6+9+10

AF 03 02 06 0B 02 05 07 08 0A 00: Same as AT+DATACUT3=11,2,5~7+8~10

0xAB downlink command can be used for set AT+SEARCHx

Example: AB aa 01 03 xx xx xx (03 here means there are total 3 bytes after 03) So

- AB aa 01 03 xx xx xx same as AT+SEARCHaa=1,xx xx xx
- AB aa 02 03 xx xx xx 02 yy yy(03 means there are 3 bytes after 03, they are xx xx xx;02 means there are 2 bytes after 02, they are yy yy) so the commands

AB aa 02 03 xx xx xx 02 yy yy same as AT+SEARCHaa=2,xx xx xx+yy yy

Fast command to handle MODBUS device

AT+MBFUN is valid since v1.3 firmware version. The command is for fast configure to read Modbus devices. It is only valid for the devices which follow the [MODBUS-RTU protocol](#).

This command is valid since v1.3 firmware version

AT+MBFUN has only two value:

- **AT+MBFUN=1:** Enable Modbus reading. And get response base on the MODBUS return

AT+MBFUN=1, device can auto read the Modbus function code: 01, 02, 03 or 04. AT+MBFUN has lower priority vs AT+DATAACUT command. If AT+DATAACUT command is configured, AT+MBFUN will be ignore.

- **AT+MBFUN=0:** Disable Modbus fast reading.

Example:

- AT+MBFUN=1 and AT+DATAACUT1/AT+DATAACUT2 are not configure (0,0,0).
- AT+COMMAND1= 01 03 00 10 00 08,1 --> read slave address 01 , function code 03, start address 00 01, quantity of registers 00 08.
- AT+COMMAND2= 01 02 00 40 00 10,1 --> read slave address 01 , function code 02, start address 00 40, quantity of inputs 00 10.

```

AT+COMMAND1=01 03 00 10 00 08,1 AT+DATAACUT1=0,0,0 AT+CMDDL1=0
AT+COMMAND2=01 02 00 40 00 10,1 AT+DATAACUT2=0,0,0 AT+CMDDL2=0
AT+COMMAND3=0,0 AT+DATAACUT3=0,0,0 AT+CMDDL3=0
AT+COMMAND4=0,0 AT+DATAACUT4=0,0,0 AT+CMDDL4=0
AT+COMMAND5=0,0 AT+DATAACUT5=0,0,0 AT+CMDDL5=0
AT+COMMAND6=0,0 AT+DATAACUT6=0,0,0 AT+CMDDL6=0
AT+COMMAND7=0,0 AT+DATAACUT7=0,0,0 AT+CMDDL7=0
AT+COMMAND8=0,0 AT+DATAACUT8=0,0,0 AT+CMDDL8=0
AT+COMMAND9=0,0 AT+DATAACUT9=0,0,0 AT+CMDDL9=0
AT+COMMANDA=0,0 AT+DATAACUTA=0,0,0 AT+CMDDLA=0
AT+COMMANDB=0,0 AT+DATAACUTB=0,0,0 AT+CMDDLB=0
AT+COMMANDC=0,0 AT+DATAACUTC=0,0,0 AT+CMDDLC=0
AT+COMMANDD=0,0 AT+DATAACUTD=0,0,0 AT+CMDDL D=0
AT+COMMAND E=0,0 AT+DATAACUTE=0,0,0 AT+CMDDLE=0
AT+COMMANDF=0,0 AT+DATAACUTF=0,0,0 AT+CMDDL F=0
    
```

Start Tx events

OK

```

CMD1 = 01 03 00 10 00 08 45 c9
RETURN1 = 01 03 10 01 00 05 ff 00 00 00 00 01 03 00 00 00 00 00 00 86 fe
CMD2 = 01 02 00 40 00 10 78 12
RETURN2 = 01 02 02 20 00 a0 78
Payload = 0C FC 01 01 00 05 ff 00 00 00 00 01 03 00 00 00 00 00 00 20 00
    
```

DATA1:8 register values

DATA2:16 register values

DATA:DATA1+DATA2

```

[177893]***** UpLinkCounter= 4 *****
[177895]TX on freq 867700000 Hz at DR 3
[178145]RX on freq 869525000 Hz at DR 3
[178148]txDone
[179143]RX on freq 867700000 Hz at DR 3
[179183]RX on freq 869525000 Hz at DR 3
[179185]rxTimeOut
    
```

- **Downlink Commands:**

A9 aa --> Same as AT+MBFUN=aa

RS485 command timeout

Some Modbus device has slow action to send replies. This command is used to configure the RS485-BL to use longer time to wait for their action.

Default value: 0, range: 0 ~ 5 seconds

- **AT Command:**

AT+CMDDLaa=hex(bb cc)

Example:

AT+CMDDL1=1000 to send the open time to 1000ms

- **Downlink Payload:**

0x AA aa bb cc Same as: **AT+CMDDLaa=hex(bb cc)**

Example:

0xAA 01 03 E8 --> Same as **AT+CMDDL1=1000 ms**

Uplink payload mode

Define to use one uplink or multiple uplinks for the sampling.

The use of this command please see: [Compose Uplink payload](#)

- **AT Command:**

AT+DATAUP=0

AT+DATAUP=1

- **Downlink Payload:**

0xAD 00 --> Same as AT+DATAUP=0

0xAD 01 --> Same as AT+DATAUP=1 //Each uplink is sent to the server one after the other as it is segmented.

- **AT Command:**

AT+DATAUP=1,Timeout

- **Downlink Payload:**

0xAD 01 00 00 14 --> Same as AT+DATAUP=1, 20000 //(00 00 14 is 20 seconds)

Each uplink is sent to the server at 20-second intervals when segmented.

Manually trigger an Uplink

Ask device to send an uplink immediately.

- **Downlink Payload:**

0x08 FF, RS485-BL will immediately send an uplink.

Clear RS485 Command

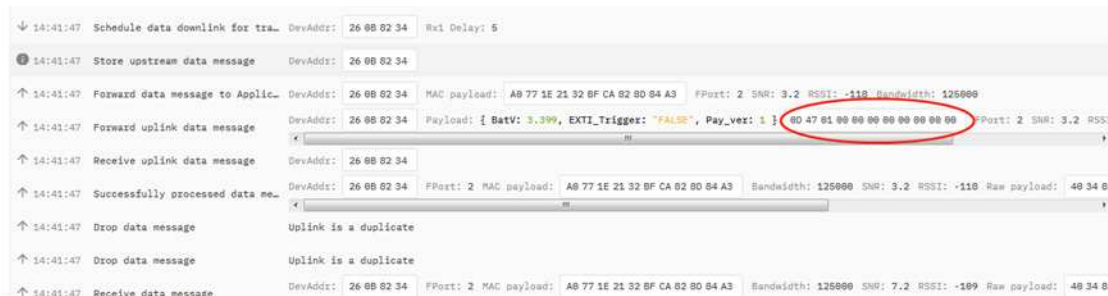
The AT+COMMANDx and AT+DATA CUTx settings are stored in special location, user can use below command to clear them.

- **AT Command:**

AT+CMDEAR=mm,nn mm: start position of erase ,nn: stop position of erase Etc. AT+CMDEAR=1,10 means erase AT+COMMAND1/AT+DATA CUT1 to AT+COMMAND10/AT+DATA CUT10

Example screen shot after clear all RS485 commands.

The uplink screen shot is:



- **Downlink Payload:**

0x09 aa bb same as AT+CMDEAR=aa,bb

Set Serial Communication Parameters

Set the Rs485 serial communication parameters:

- **AT Command:**

- **Set Baud Rate:**

AT+BAUDR=9600 // Options: (200~115200) , When using low baud rate or receiving multiple bytes, you need to use AT+CMDDL to increase the receive timeout (the default receive timeout is 300ms), otherwise data will be lost

- **Set UART Parity**

AT+PARITY=0 // Option: 0: no parity, 1: odd parity, 2: even parity

- **Set STOPBIT**

AT+STOPBIT=0 // Option: 0 for 1bit; 1 for 1.5 bit ; 2 for 2 bits

- **Downlink Payload:**

A7 01 aa bb: Same AT+BAUDR=hex(aa bb)*100

Example:

- A7 01 00 60 same as AT+BAUDR=9600
- A7 01 04 80 same as AT+BAUDR=115200

A7 02 aa: Same as AT+PARITY=aa (aa value: 00 , 01 or 02)

A7 03 aa: Same as AT+STOPBIT=aa (aa value: 00 , 01 or 02)

Configure Databit(Since version 1.4.0)

- **AT Command:**

AT+DATABIT=7 // Set the data bits to 7

AT+DATABIT=8 //Set the data bits to 8

- **Downlink Payload:**

A7 04 07: Same as AT+DATABIT=7

A7 04 08: Same as AT+DATABIT=8

Encrypted payload

- **AT Command:**

AT+DECRYPT=1 // The payload is uploaded without encryption

AT+DECRYPT=0 // Encrypt when uploading payload (default)

Get sensor value

- **AT Command:**

AT+GETSENSORVALUE=0 // The serial port gets the reading of the current sensor

AT+GETSENSORVALUE=1 // The serial port gets the current sensor reading and uploads it.

Resets the downlink packet count

- **AT Command:**

AT+DISFCNTCHECK=0 // When the downlink packet count sent by the server is less than the node downlink packet count or exceeds 16384, the node will no longer receive downlink packets (default)

AT+DISFCNTCHECK=1 // When the downlink packet count sent by the server is less than the node downlink packet count or exceeds 16384, the node resets the downlink packet count and keeps it consistent with the server downlink packet count.

When the limit bytes are exceeded, upload in batches

- **AT Command:**

AT+DISMACANS=0 // When the MACANS of the reply server plus the payload exceeds the maximum number of bytes of 11 bytes (DR0 of US915, DR2 of AS923, DR2 of AU195), the node will send a packet with a payload of 00 and a port of 4. (default)

AT+DISMACANS=1 // When the MACANS of the reply server plus the payload exceeds the maximum number of bytes of the DR, the node will ignore the MACANS and not reply, and only upload the payload part.

- **Downlink Payload**

0x21 00 01 // Set the DISMACANS=1

Copy downlink to uplink

- **AT Command:**

AT+RPL=5 // After receiving the package from the server, it will immediately upload the content of the package to the server, the port number is 100.

Example: **aa xx xx xx xx** // aa indicates whether the configuration has changed, 00 is yes, 01 is no; xx xx xx xx are the bytes sent.

The screenshot shows the 'Live data' tab with a table of data messages. The first message at 14:22:12 is a 'Forward uplink data message' with DevAddr: 26 08 7C A7 and Payload: { 00 11 22 33 44 55 66 77 }. The second message at 14:22:12 is 'Successfully processed data ...' with DevAddr: 26 08 7C A7. The interface includes controls for 'Verbose stream', 'Export as JSON', 'Pause', and 'Clear'.

For example, sending 11 22 33 44 55 66 77 will return invalid configuration 00 11 22 33 44 55 66 77.

The screenshot shows the 'Live data' tab with a table of data messages. The first message at 14:26:13 is a 'Forward uplink data message' with DevAddr: 26 08 7C A7 and Payload: { 01 01 00 02 58 }. The second message at 14:26:13 is 'Successfully processed data ...' with DevAddr: 26 08 7C A7. The interface includes controls for 'Verbose stream', 'Export as JSON', 'Pause', and 'Clear'.

For example, if 01 00 02 58 is issued, a valid configuration of 01 01 00 02 58 will be returned.

Query version number and frequency band \ TDC

- **Downlink Payload: 26 01** // Downlink 26 01 can query device upload frequency, frequency band, software version number, TDC time.

Example:

The screenshot shows the 'Live data' tab with a table of data messages. The message at 16:08:08 is a 'Forward uplink data message' with Payload: { FIRMWARE_VERSION: "1.6.0", FREQUENCY_BAND: "EU868", SUB_BAND: "NULL", TDC_sec: 600 } and DevAddr: 26 01 FF 01 60 00 02 58. The interface includes controls for 'Verbose stream', 'Export as JSON', 'Pause', and 'Clear'.

Control output power duration

User can set the output power duration before each sampling.

- **AT Command:**

Example:

AT+3V3T=1000 // 3V3 output power will open 1s before each sampling.

AT+5VT=1000 // +5V output power will open 1s before each sampling.

- **LoRaWAN Downlink Command:**

07 01 aa bb Same as AT+5VT=(aa bb)

07 02 aa bb Same as AT+3V3T=(aa bb)

3.6 Buttons

Button	Feature
RST	Reboot RS485-BL

3.7 +3V3 Output (Since v1.3.3)

RS485-BL has a Controllable +3V3 output, user can use this output to power external sensor.

The +3V3 output will be valid for every sampling. RS485-BL will enable +3V3 output before all sampling and disable the +3V3 after all sampling.

The +3V3 output time can be controlled by AT Command.

AT+3V3T=1000

Means set +3v3 valid time to have 1000ms. So, the real +3v3 output will actually have 1000ms + sampling time for other sensors.

By default, the AT+3V3T=0. This is a special case, means the +3V3 output is always on at any time

3.8 +5V Output (Since v1.3.3)

RS485-BL has a Controllable +5V output, user can use this output to power external sensor.

The +5V output will be valid for every sampling. RS485-BL will enable +5V output before all sampling and disable the +5v after all sampling.

The 5V output time can be controlled by AT Command.

(AT+5VT increased from the maximum 5000ms to 65000ms. Since v1.4.0)

AT+5VT=1000

Means set 5V valid time to have 1000ms. So, the real 5V output will actually have 1000ms + sampling time for other sensors.

By default, the AT+5VT=0. If the external sensor which require 5v and require more time to get stable state, user can use this command to increase the power ON duration for this sensor.

3.9 LEDs

LEDs	Feature
LED1	Blink when device transmit a packet.

3.10 Switch Jumper

Switch Jumper	Feature
SW1	ISP position: Upgrade firmware via UART Flash position: Configure device, check running status.
SW2	5V position: set to compatible with 5v I/O. 3.3v position: set to compatible with 3.3v I/O.,

+3.3V: is always ON

+5V: Only open before every sampling. The time is by default, it is **AT+5VT=0**. Max open time. 65000 ms. (Since v1.4.0)

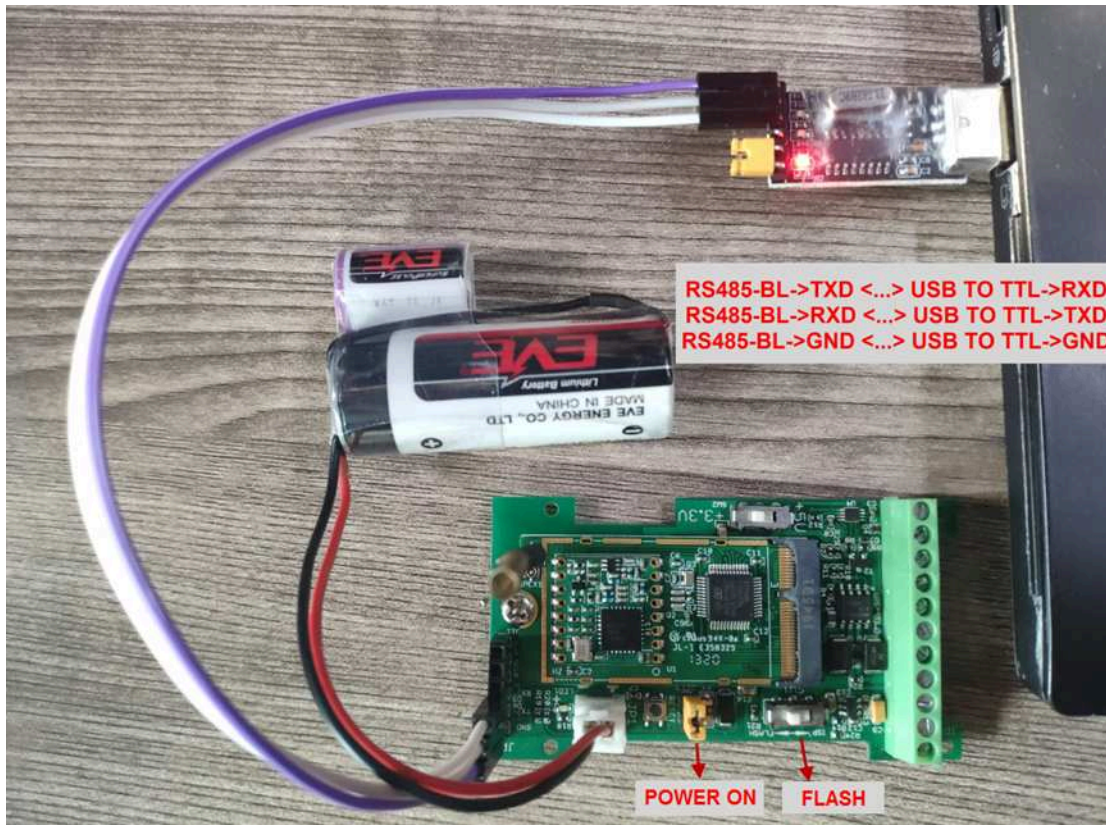
4. Case Study

User can check this URL for some case studies: [APP RS485 COMMUNICATE WITH SENSORS](#)

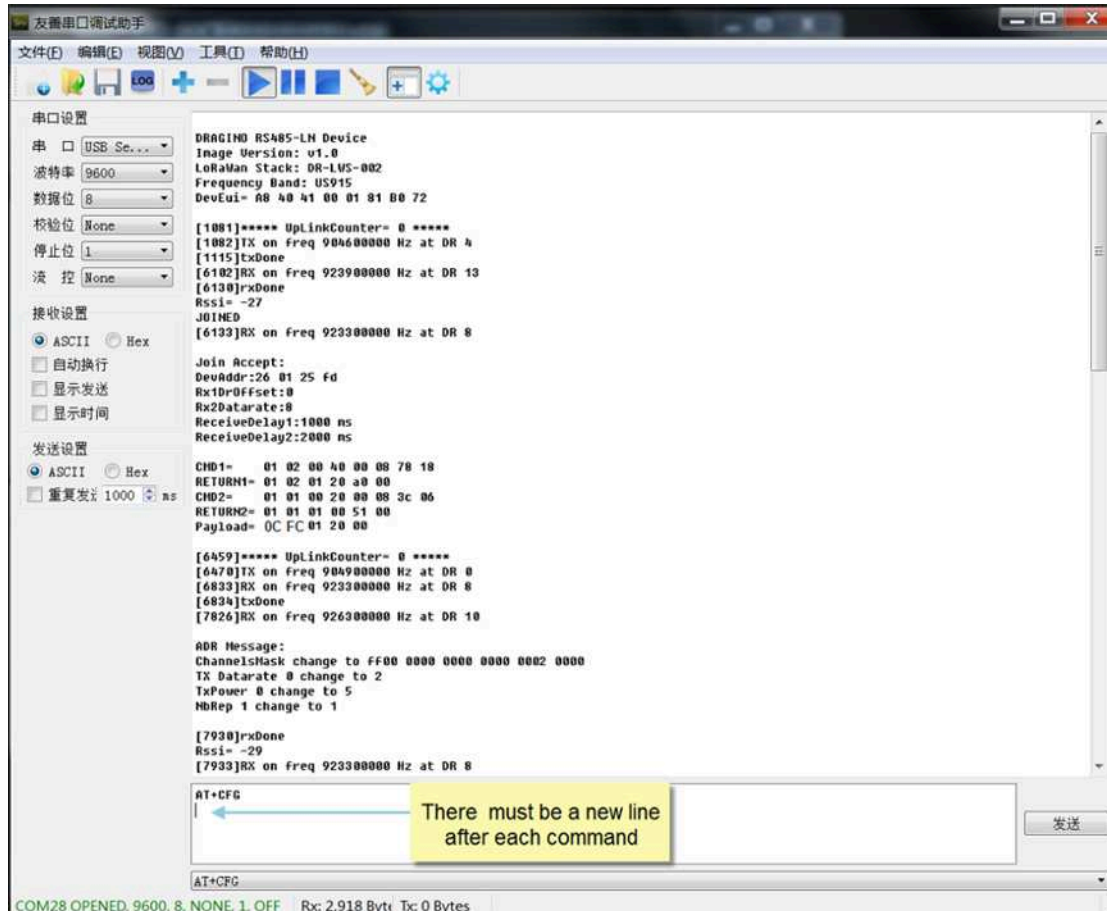
5. Use AT Command

5.1 Access AT Command

RS485-BL supports AT Command set. User can use a USB to TTL adapter plus the 3.5mm Program Cable to connect to RS485-BL to use AT command, as below.



In PC, User needs to set **serial tool**(such as [putty](#), SecureCRT) baud rate to **9600** to access to access serial console of RS485-BL. The default password is 123456. Below is the output for reference:



More detail AT Command manual can be found at [AT Command Manual](#)

5.2 Common AT Command Sequence

5.2.1 Multi-channel ABP mode (Use with SX1301/LG308)

If device has not joined network yet:

- **AT+FDR**
- **AT+NJM=0**
- **ATZ**

If device already joined network:

- **AT+NJM=0**
- **ATZ**

5.2.2 Single-channel ABP mode (Use with LG01/LG02)

AT+FDR Reset Parameters to Factory Default, Keys Reserve

AT+NJM=0 Set to ABP mode

AT+ADR=0 Set the Adaptive Data Rate Off

AT+DR=5 Set Data Rate

AT+TDC=60000 Set transmit interval to 60 seconds

AT+CHS=868400000 Set transmit frequency to 868.4Mhz

AT+RX2FQ=868400000 Set RX2Frequency to 868.4Mhz (according to the result from server)

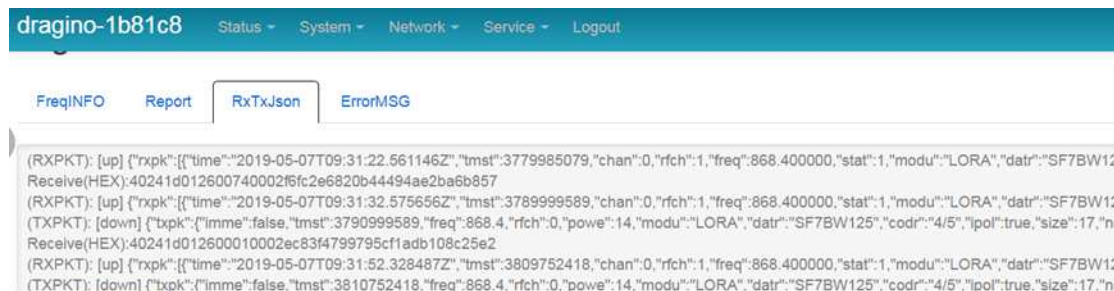
AT+RX2DR=5 Set RX2DR to match the downlink DR from server. see below

AT+DADDR=26 01 1A F1 Set Device Address to 26 01 1A F1, this ID can be found in the LoRa Server portal.

ATZ Reset MCU

Note:

1. Make sure the device is set to ABP mode in the IoT Server.
2. Make sure the LG01/02 gateway RX frequency is exactly the same as AT+CHS setting.
3. Make sure SF / bandwidth setting in LG01/LG02 match the settings of AT+DR. refer [this link](#) to see what DR means.
4. The command AT+RX2FQ and AT+RX2DR is to let downlink work. to set the correct parameters, user can check the actually downlink parameters to be used. As below. Which shows the RX2FQ should use 868400000 and RX2DR should be 5



6. FAQ

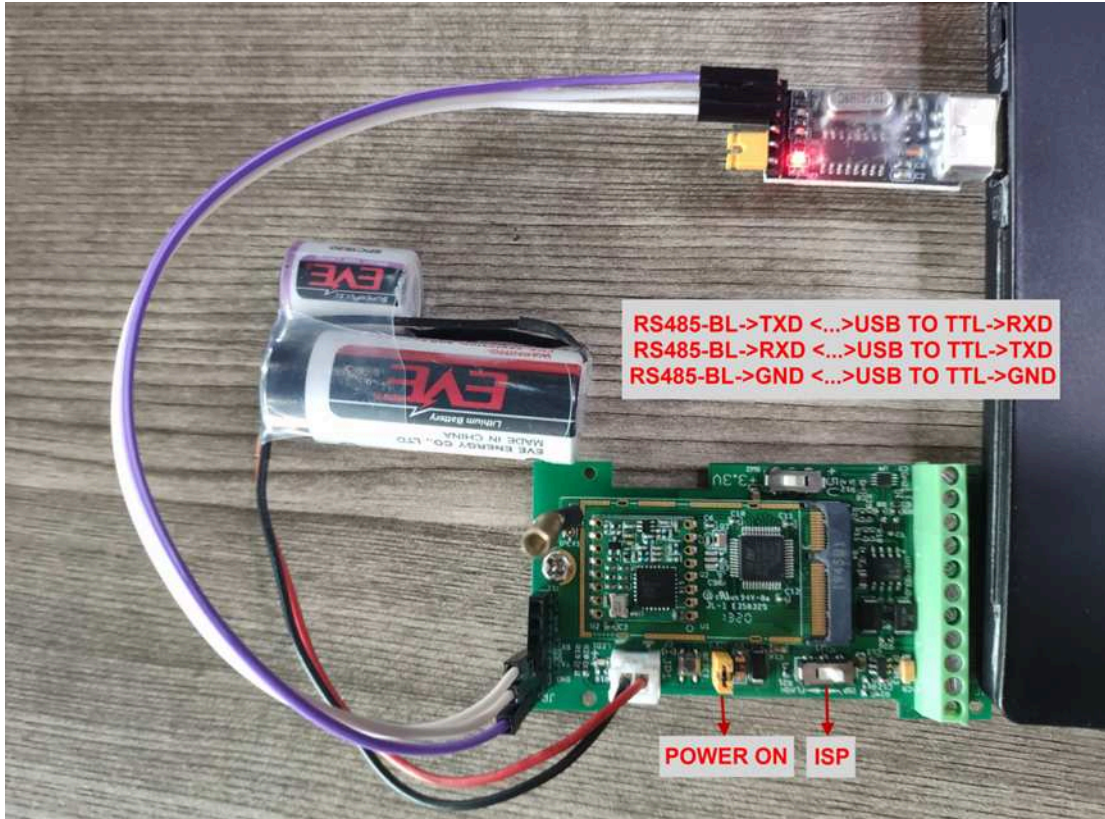
6.1 How to upgrade the image?

The RS485-BL LoRaWAN Controller is shipped with a 3.5mm cable, the cable is used to upload image to RS485-BL to:

- Support new features
- For bug fix
- Change LoRaWAN bands.

Below shows the hardware connection for how to upload an image to RS485-BL:

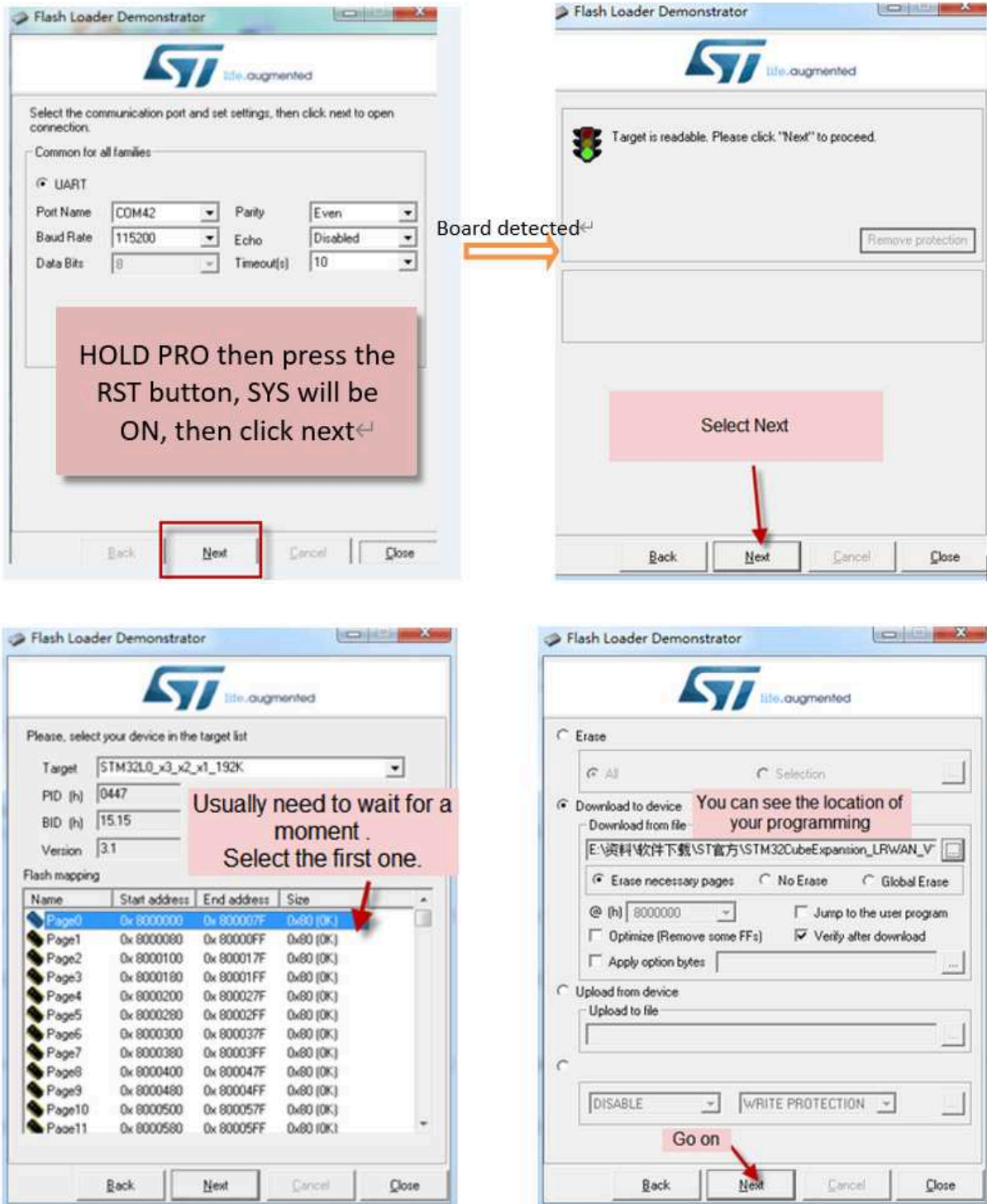




Step1: Download [flash loader](#).

Step2: Download the [LT Image files](#).

Step3: Open flashloader; choose the correct COM port to update.





6.2 How to change the LoRa Frequency Bands/Region?

User can follow the introduction for [how to upgrade image](#). When download the images, choose the required image file for download.

6.3 How many RS485-Slave can RS485-BL connects?

The RS485-BL can support max 32 RS485 devices. Each uplink command of RS485-BL can support max 16 different RS485 command. So RS485-BL can support max 16 RS485 devices pre-program in the device for uplink. For other devices no pre-program, user can use the [downlink message \(type code 0xA8\) to poll their info](#).

6.4 How to Use RS485-BL to connect to RS232 devices?

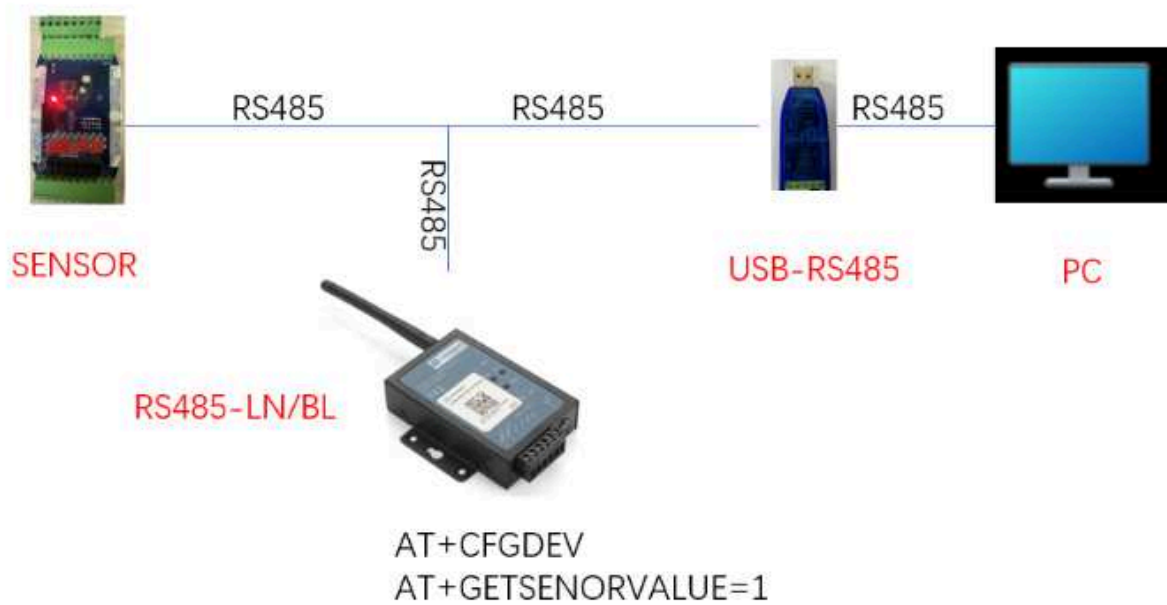
[Use RS485-BL or RS485-LN to connect to RS232 devices. - DRAGINO](#)

6.5 How to judge whether there is a problem with the set COMMAND

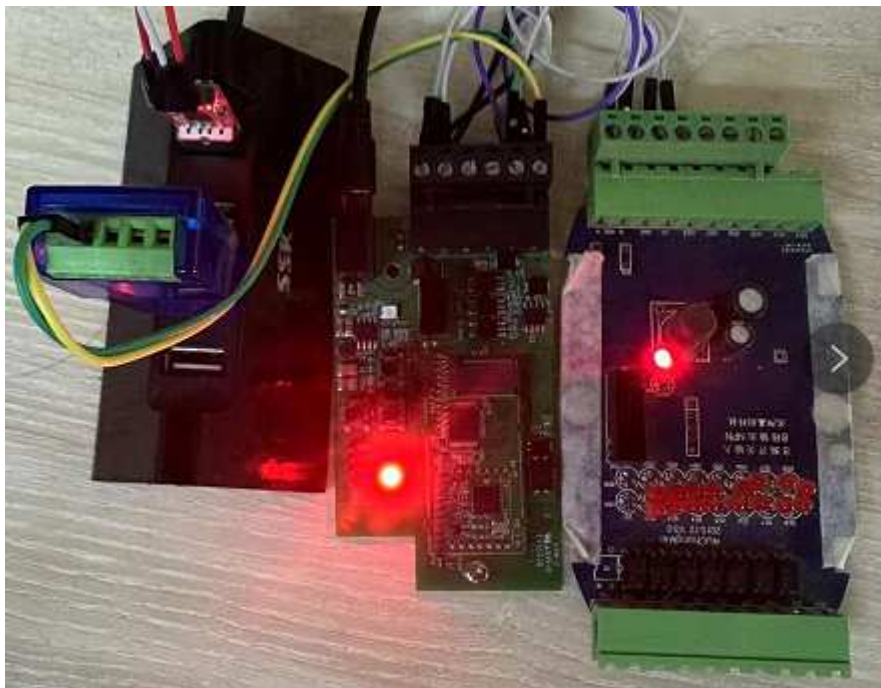
6.5.1 Introduce:

Users can use below the structure to fast debug the communication between RS485BL and RS485-LN. The principle is to put the PC in the RS485 network and sniff the packet between Modbus MTU and RS485-BL/LN. We can [use this way to](#):

1. Test if Modbus-MTU works with PC commands.
2. Check if RS485-LN sent the expected command to Mobus-MTU
3. Check if Modbus-MTU return back the expected result to RS485-LN.
4. If both b) and c) has issue, we can compare PC's output and RS485-LN output.

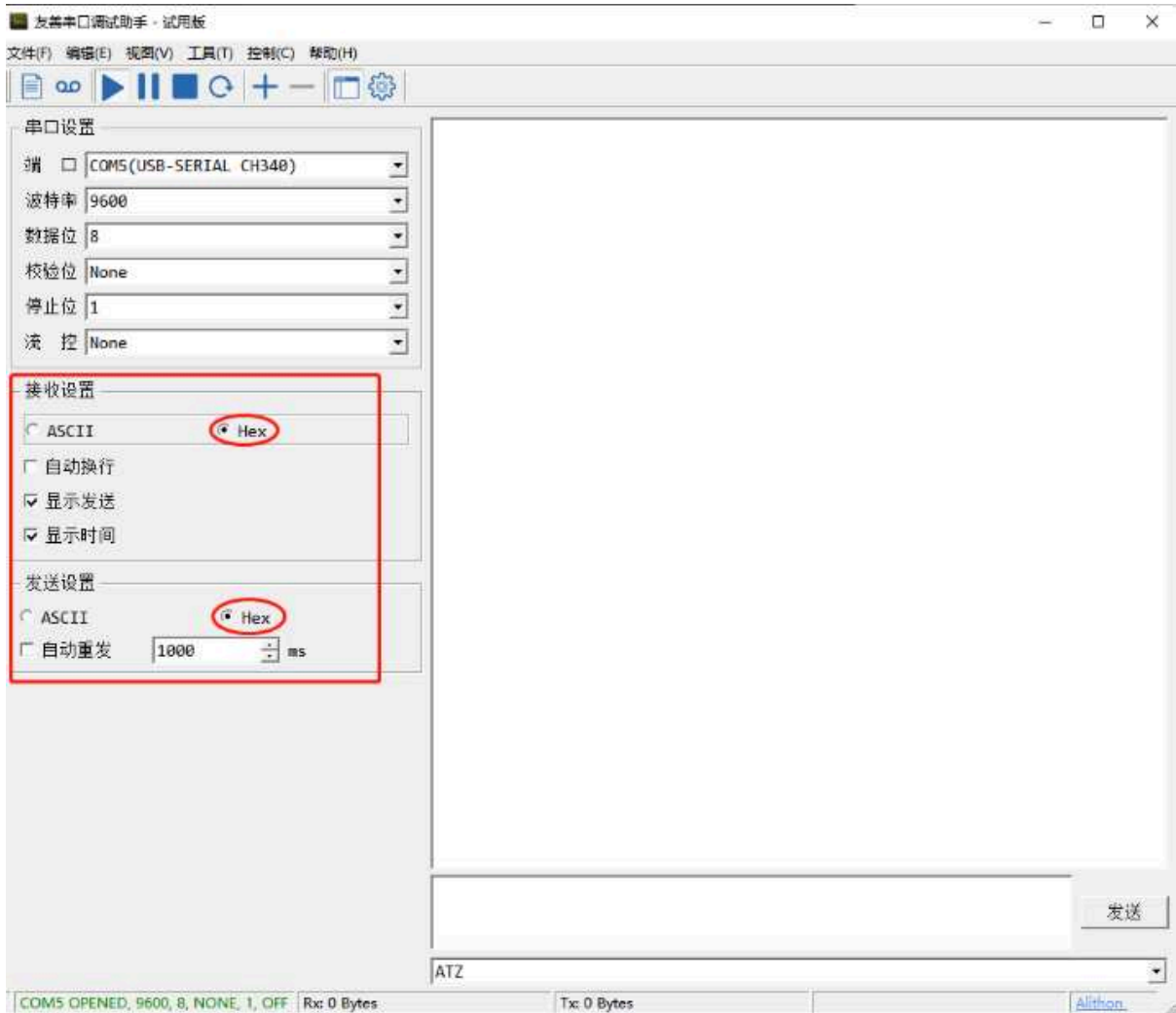


Example Connection:



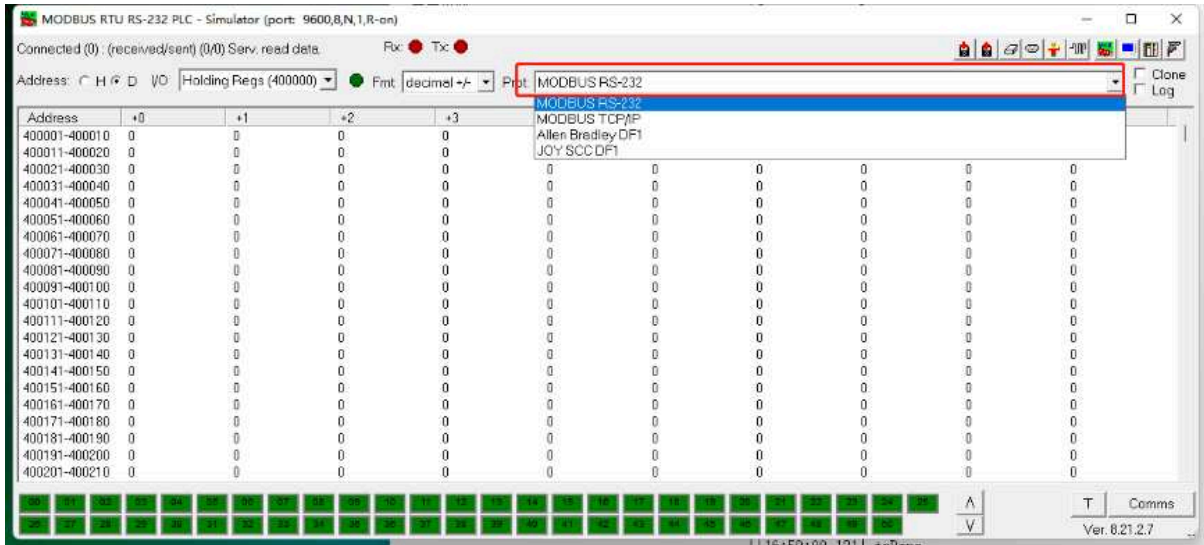
6.5.2 Set up PC to monitor RS485 network With Serial tool

Note: Receive and send set to hex mode

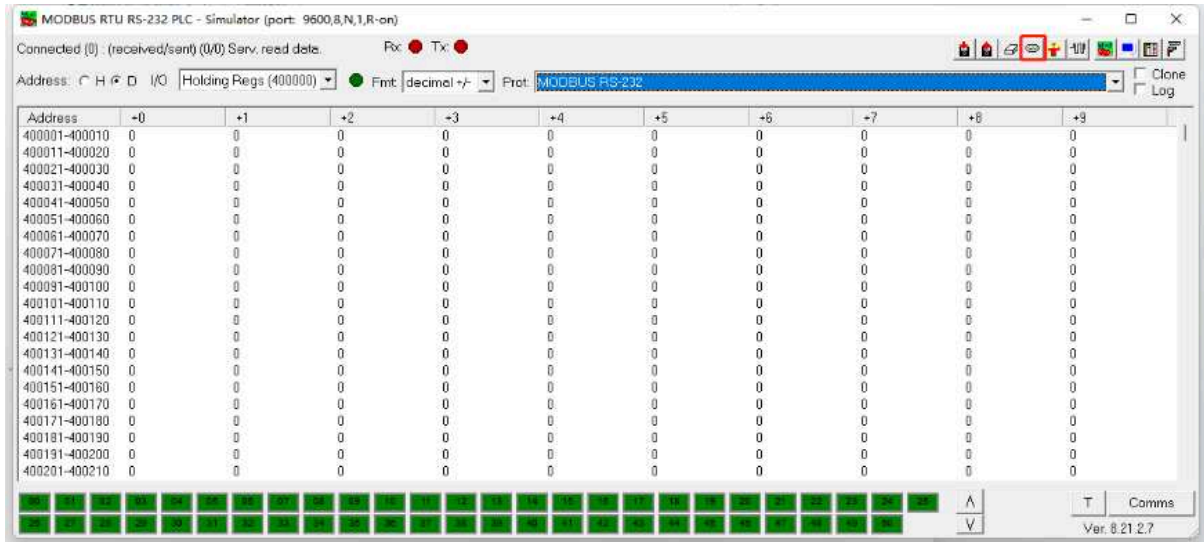


6.5.3 With ModRssim2:

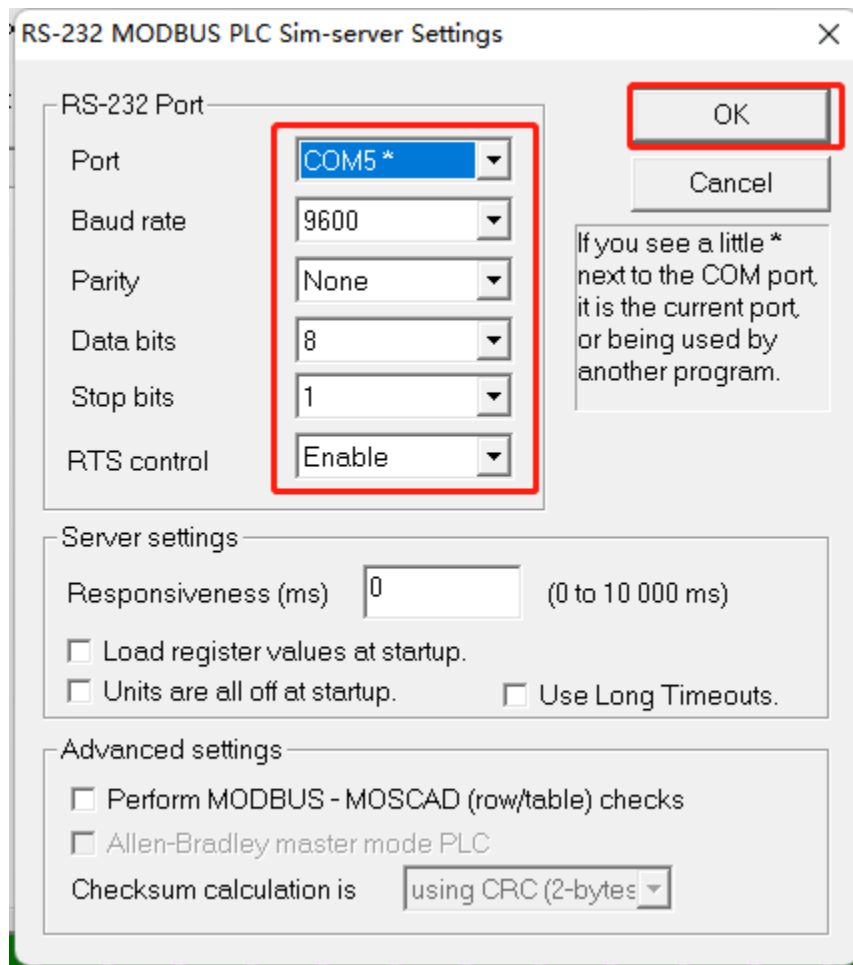
- (1) Select serial port MODBUS RS-232



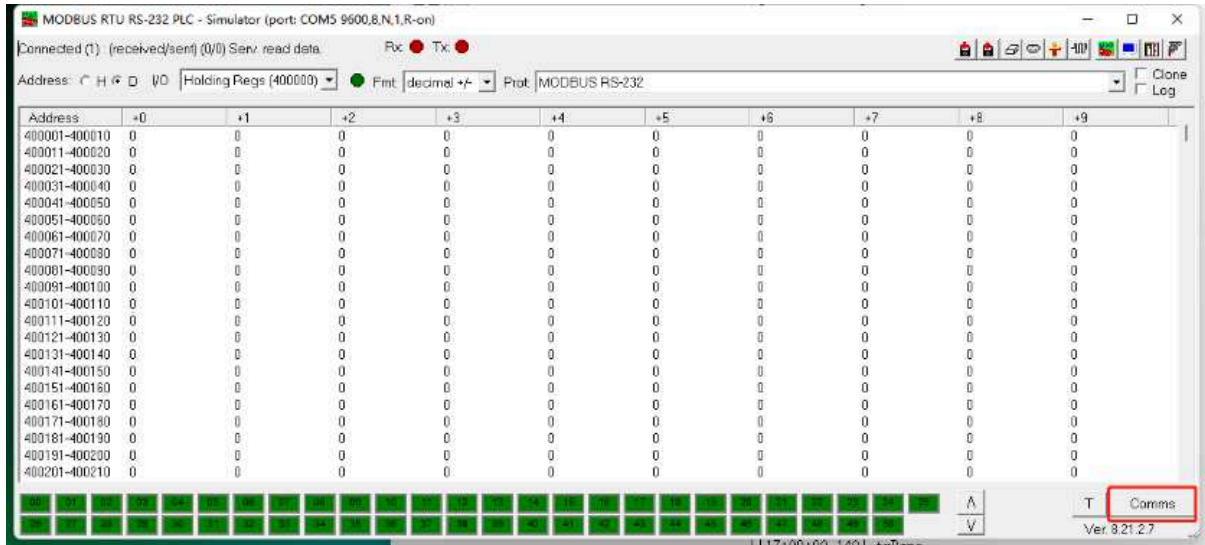
(2) Click the serial port icon



(3) After selecting the correct serial port and baud rate, click ok



(4) Click the comms.



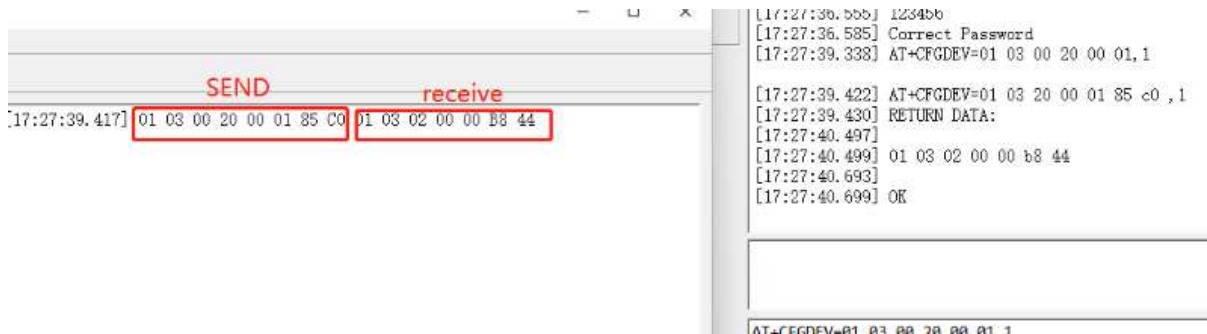
Run RS485-LN/BL command and monitor if it is correct.

6.5.4 Example – Test the CFGDEV command

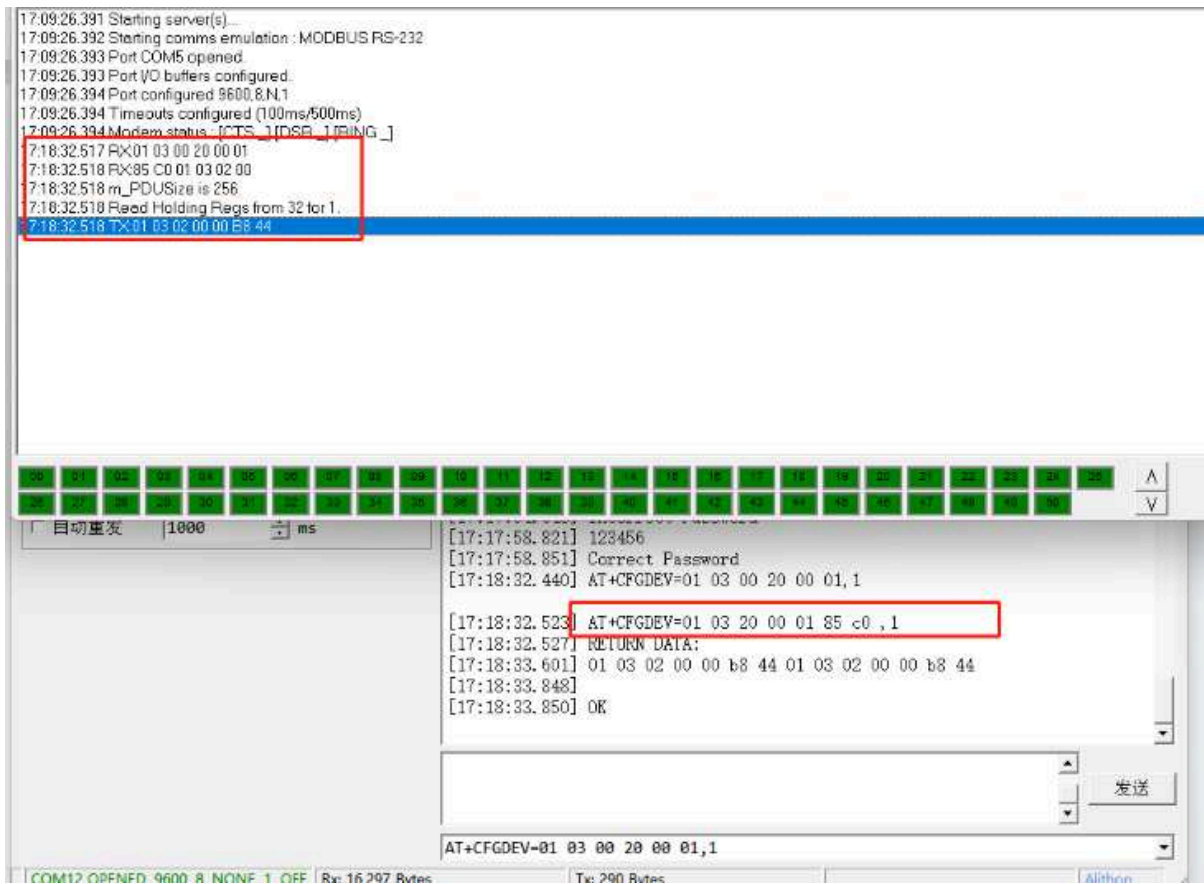
RS485-LN sent below command:

AT+CFGDEV=01 03 20 00 01 85 c0,1 to RS485 network, and PC is able to get this command and return commands from MTU to show in the serial tool.

We can see the output from the Serial port tool to analyze. And check if they are expected result.



We can also use ModRSsim2 to see the output.



6.5.5 Example – Test CMD command sets.

Run **AT+SENSORVALUE=1** to test the CMD commands set in RS485-LN.

Serial port tool:


```
[17:27:39.417] 01 03 00 20 00 01 85 C0 01 03 02 00 00 B8 44 01 03 00 20 00
01 85 C0 01 03 02 00 00 B8 44 01 03 00 21 00 01 D4 00 01 03 02 00 00 B8 44
01 03 00 22 00 01 24 00 01 03 02 00 20 B9 9C 01 03 00 23 00 01 75 C0 01 03
02 00 00 B8 44 01 03 00 24 00 01 C4 01 01 83 01 80 F0

[17:27:40.699] OK
[17:30:01.230] AT+GETSENSORVALUE=1

[17:30:01.260]
[17:30:01.261] OK
[17:30:03.349]
[17:30:03.351] CMD1 = 01 03 00 20 00 01 85 c0
[17:30:03.401]
[17:30:03.402] RETURN1 = 01 03 02 00 00 b8 44
[17:30:03.440]
[17:30:03.441] CMD2 = 01 03 00 21 00 01 d4 00
[17:30:03.487]
[17:30:03.491] RETURN2 = 01 03 02 00 00 b8 44
[17:30:03.532] CMD3 = 01 03 00 22 00 01 24 00
[17:30:03.571] RETURN3 = 01 03 02 00 20 b9 9c
[17:30:03.611]
[17:30:03.611] CMD4 = 01 03 00 23 00 01 75 c0
[17:30:03.661] RETURN4 = 01 03 02 00 00 b8 44
[17:30:03.700]
[17:30:03.702] CMD5 = 01 03 00 24 00 01 c4 01
[17:30:03.751]
[17:30:03.751] RETURN5 = 01 83 01 80 f0 00 00
[17:30:03.779]
```

ModRssim2:

```
17:24:36.080 m_PDUSize is 256
17:24:36.080 Read Holding Fiegs from 35 for 1.
17:24:36.080 TX:01 03 02 00 00 B8 44
17:24:36.991 RX:01 03 00 23 00 01
17:24:36.991 RX:75 C0 01 03 02 00
17:24:36.991 m_PDUSize is 256
17:24:36.991 Read Holding Fiegs from 35 for 1.
17:24:36.992 TX:01 03 02 00 00 B8 44
17:24:37.901 RX:01 03 00 24 00 01
17:24:37.901 RX:C4 01 01 83 01 80
17:24:37.902 m_PDUSize is 256
17:24:37.902 Read Holding Fiegs from 36 for 1.
17:24:37.902 TX:01 03 02 00 00 B8 44
17:24:38.576 RX:01 03 00 24 00 01
17:24:38.577 RX:C4 01 01 83 01 80
17:24:38.577 m_PDUSize is 256
17:24:38.577 Read Holding Fiegs from 36 for 1.
17:24:38.577 TX:01 03 02 00 00 B8 44
17:24:39.480 RX:01 03 00 24 00 01
17:24:39.480 RX:C4 01 01 83 01 80
17:24:39.480 m_PDUSize is 256
17:24:39.480 Read Holding Fiegs from 36 for 1.
17:24:39.480 TX:01 03 02 00 00 B8 44

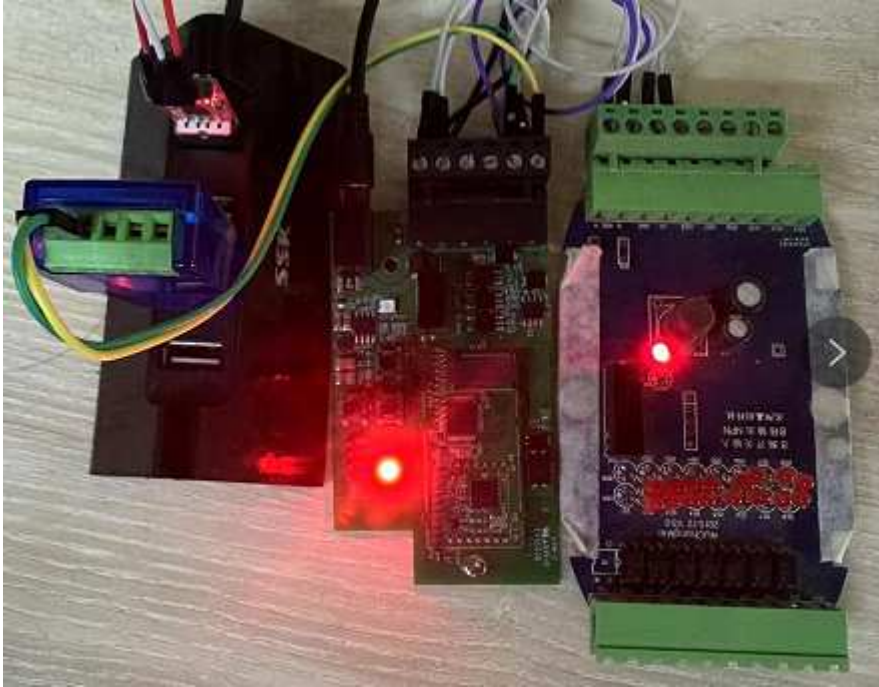
[17:24:27.387]
[17:24:27.387] OK
[17:24:40.362]
[17:24:40.395]
[17:24:40.397] CMD1 = 01 03 00 20 00 01 85 c0
[17:24:40.446]
[17:24:40.447] RETURN1 = 01 03 02 00 00 b8 44
[17:24:40.487] CMD2 = 01 03 00 21 00 01 d4 00
[17:24:40.537] RETURN2 = 01 03 02 00 00 b8 44
[17:24:40.575]
[17:24:40.577] CMD3 = 01 03 00 22 00 01 24 00
[17:24:40.617] RETURN3 = 01 03 02 00 20 b9 9c
[17:24:40.665]
[17:24:40.667] CMD4 = 01 03 00 23 00 01 75 c0
[17:24:40.704] RETURN4 = 01 03 02 00 00 b8 44
[17:24:40.707] CMD5 = 01 03 00 24 00 01 c4 01
[17:24:40.745] RETURN5 = 01 83 01 80 f0 01 03
[17:24:40.794]
[17:24:40.797] RETURNS = 01 83 01 80 f0 01 03
```

6.5.6 Test with PC

If there is still have problem to set up correctly the commands between RS485-LN and MTU. User can test the correct RS485 command set in PC and compare with the RS485 command sent out via RS485-LN. as long as both commands are the same, the MTU should return correct result.

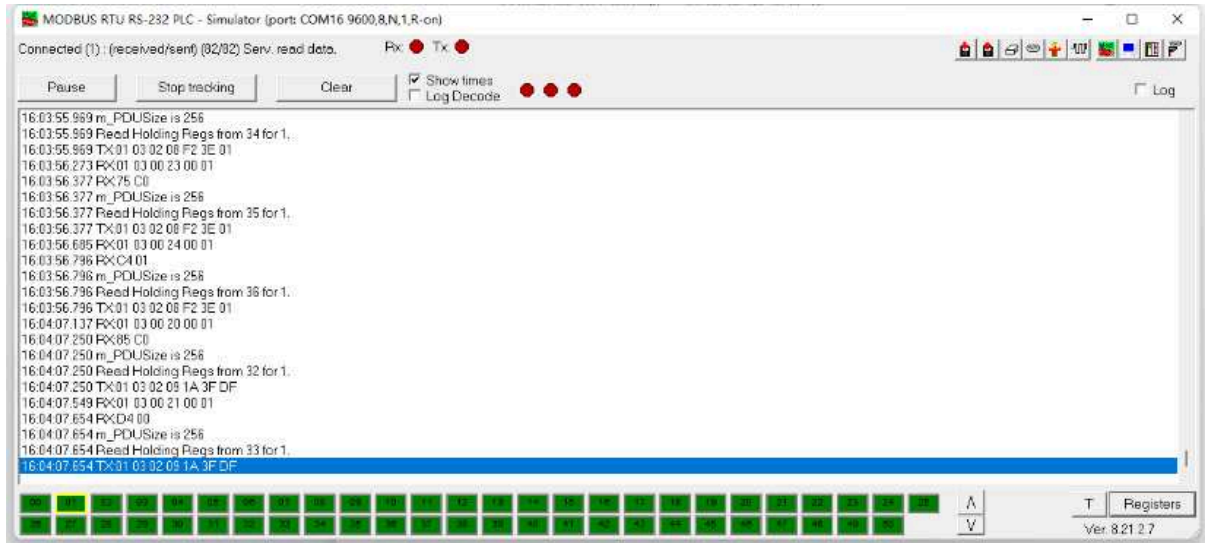
Or User can send the working commands set in PC serial tool to Dragino Support to check what should be configured in RS485-LN.

Connection method:



Link situation:

```
[16:03:35.274] CMD4 = 01 03 00 23 00 01 75 c0
[16:03:35.320] RETURN4 = 01 03 02 08 98 be 2e
[16:03:35.367] CMD5 = 01 03 00 24 00 01 c4 01
[16:03:35.412] RETURN5 = 01 03 02 08 98 be 2e
[16:03:35.444] Payload = 01
[16:03:35.951]
[16:03:35.961] [134251]***** UpLinkCounter= 11 *****
[16:03:35.992] [134252]TX on freq 910700000 Hz at DR 0
[16:03:36.245] [134545]RX on freq 923300000 Hz at DR 8
[16:03:36.287] [134547]txDone
[16:03:41.239] [139539]RX on freq 924500000 Hz at DR 10
[16:03:41.281] [139571]RX on freq 923300000 Hz at DR 8
[16:03:41.311] [139573]rxTimeOut
[16:03:46.016]
[16:03:46.019] CMD1 = 01 03 00 20 00 01 85 c0
[16:03:46.065] RETURN1 = 01 03 02 08 c0 bf d4 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00
[16:03:46.204] CMD2 = 01 03 00 21 00 01 d4 00
[16:03:46.252] RETURN2 = 01 03 02 08 c0 bf d4
[16:03:46.297] CMD3 = 01 03 00 22 00 01 24 00
[16:03:46.344] RETURN3 = 01 03 02 08 c0 bf d4
[16:03:46.374] CMD4 = 01 03 00 23 00 01 75 c0
[16:03:46.421] RETURN4 = 01 03 02 08 c0 bf d4
[16:03:46.467] CMD5 = 01 03 00 24 00 01 c4 01
[16:03:46.515] RETURN5 = 01 03 02 08 c0 bf d4
[16:03:46.545] Payload = 01
[16:03:47.052]
[16:03:47.057] [145352]***** UpLinkCounter= 12 *****
[16:03:47.088] [145354]TX on freq 910500000 Hz at DR 0
[16:03:47.347] [145646]RX on freq 923300000 Hz at DR 8
[16:03:47.383] [145648]txDone
[16:03:52.340] [150639]RX on freq 923900000 Hz at DR 10
[16:03:52.376] [150671]RX on freq 923300000 Hz at DR 8
[16:03:52.407] [150673]rxTimeOut
```



6.6 Where to get the decoder for RS485-BL?

The decoder for RS485-BL needs to be written by yourself. Because the sensor to which the user is connected is custom, the read device data bytes also need custom parsing, so there is no universal decoder. We can only provide [templates](#) for decoders (no intermediate data parsing part involved)

7. Trouble Shooting

7.1 Downlink doesn't work, how to solve it?

Please see this link for debug: [LoRaWAN Communication Debug](#)

7.2 Why I can't join TTN V3 in US915 /AU915 bands?

It might about the channels mapping. Please see for detail: [Notice of Frequency band](#)

8. Order Info

Part Number: RS485-BL-XXX

XXX:

- **EU433:** frequency bands EU433
- **EU868:** frequency bands EU868
- **KR920:** frequency bands KR920
- **CN470:** frequency bands CN470

- **AS923**: frequency bands AS923
- **AU915**: frequency bands AU915
- **US915**: frequency bands US915
- **IN865**: frequency bands IN865
- **RU864**: frequency bands RU864
- **KZ865**: frequency bands KZ865

9. Packing Info

Package Includes:

- RS485-BL x 1
- Stick Antenna for LoRa RF part x 1
- Program cable x 1

Dimension and weight:

- Device Size: 13.5 x 7 x 3 cm
- Device Weight: 105g
- Package Size / pcs : 14.5 x 8 x 5 cm
- Weight / pcs : 170g

10. Support

- Support is provided Monday to Friday, from 09:00 to 18:00 GMT+8. Due to different timezones we cannot offer live support. However, your questions will be answered as soon as possible in the before-mentioned schedule.
- Provide as much information as possible regarding your enquiry (product models, accurately describe your problem and steps to replicate it etc) and send a mail to support@dragino.com

The logo for DIREKTRONIK is displayed in a large, bold, red font with a white outline. The letters are stylized and slanted to the right, giving it a dynamic and modern appearance.