# Communication Protocol Between MR7901 and Platform

V1.12

（2020.12.08）

**SHENZHEN MARKTRACE CO.,LTD**

# Index


1. **Change Doc Record** ..... 2
2. Purpose ..... 3
3. Normative References ..... 3
4. Required instructions ..... 3
5. Communication process ..... 4
6. Communication packet format ..... 6
   - 6.1Packet format ..... 6
   - 6.2message headerMessaget header ..... 6
   - 6.3message contentMessage content ..... 9
7. Data interaction ..... 10
   - 7.1 Registered （0x0008/ 0x8008） ..... 10
   - 7.2 Logon （0x0001/ 0x8001） ..... 13
   - 7.3 Heartbeat（0x0003/ 0x8003） ..... 16
   - 7.4 Data reporting （0x0004/ 0x8004） ..... 21
   - 7.5 Firmware update （0x000D/ 0x800D） ..... 23
   - 7.6 Configuration parameters （0x000A/ 0x800A） ..... 31
   - 7.7 Tag massage （0x0009/ 0x8009） ..... 66
8. TLVindex ..... 78
   - 8.1 TLVType list and format ..... 78
   - 8.2 TLV--Tag format description ..... 79
   - 8.3 TLV--Attendance tag format description ..... 83
   - 8.4 TLV--Assets management tag format description ..... 86
   - 8.5 TLV--MR1054X tag format description ..... 87
9. Configuration Parameter Format Description ..... 88
10. Command List ..... 92
11. Check algorithm ..... 93
    - 11.1 CRC16Check algorithm ..... 93
    - 11.2 Checksum algorithm ..... 96
12. Appendix ..... 98
    - 12.1 The relationship between the battery charge and the battery voltage ..... 98
13. Security Mechanism ..... 99
    - 13.1 Data interaction security between device and platform ..... 99

# 1. Change Document Record

| SN | Ver | Data | Changes | Compiler |
|---|---|---|---|---|
| 1 | Org | 2020.04.08 | Original To V1.12 | Chuangbao Wang |
| 2 | V1.12 | 2020.06.13 | English Version | Edward Lee |

# 2. Purpose

Provide a reference for engineers to complete the communication between MR7901 and platform/server/end devices. (this protocol suites for all MR79xx products, the sample model in this document is MR7901)

# 3. **Normative References**

<Common access standard for industrial IOT equipment>.--SHENZHEN MARKTRACE CO.,LTD

# 4. Required instructions

1) MR7901 and platform direct communication methods are: GPRS, LAN or others；

2) Server: Platform software(This includes "load balancing server" and "load server")；

3) Terminal / Device: MR7901；

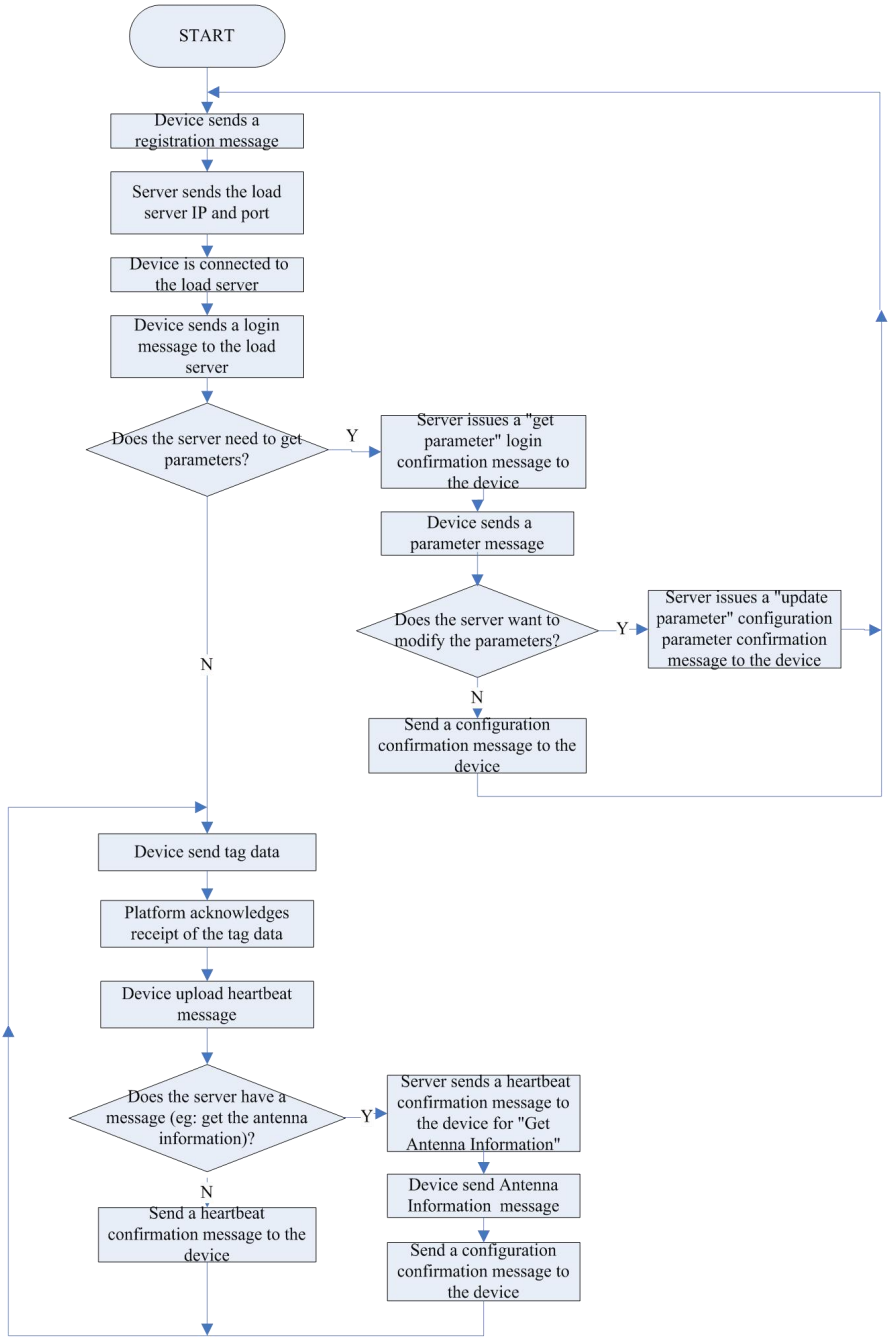
4) Load Balancing Server: Registered server. The server that receive registration message from MR7901；

5) Load server: The server receive messages from MR7901 such as login, heartbeat, data, configuration, etc.

# 5. Communication process

The communication uses the "client" - "server" mode, Platform is server and MR7901 is client. The communication is initiated by the client to the server.

After MR7901 power on, you need to submit a registration application to the platform. After the registration is successful, you can log in to the platform. After the login is successful, the data can be exchanged. Data interaction is initiated by the device, the platform responds. The following is a business process diagram:




Examples of communication processes

# 6.Communication packet format

## 6.1Packet format

The communication packet between the terminal and the platform is composed of four parts: the Start Flag, the Message Header, the Service Content and the Check. The start flag is fixed to 0x55, 0xAA.The check is the CRC16 check of the header and the service content,

The format is as follows

| Start Flag | Message Header | Service Content | Check |
|---|---|---|---|
| 2 Bytes | 28 Bytes | Variable length | 2 Bytes |

explanation ：

1. Start Flag: fixed to 0x55,0xAA
2. The length of the packet header is fixed to 28Bytes;
3. The length of the Service Content is variable;
4. Check is in accordance with CRC16 CCITT standard -0x1021 (initial value is 0xFFFF),check algorithm is described in the following sections
5. Check is the calculation of the header and the service content part
6. If encrypted communication is used, only the message content is encrypted (for encryption-related instructions, see Chapter 13 Security Mechanism).

## 6.2Message headerMessage header

The packet header is composed of message length, command code, protocol version, terminal serial number (or device ID).The header format is as follows:

| Message Header (28 Bytes) | | | |
|---|---|---|---|
| SN | Field | Length (byte) | Description |
| 1 | Total message length | 2 | Number of bytes with header and content (excluding start identifier and check) |
| 2 | Command code | 2 | Indicates the command to be executed or answered, such as login, data reporting, update, and so on. |
| 3 | Message serial number | 4 | 0x00000000 to 0xFFFFFFFF，The sender to maintain their own serial number, each successful communication, automatically add 1. after 0xFFFFFFFF，change to 0x00000000。 |
| 4 | Message protocol version | 2 | Note:<br>V2.0 (dedicated to MR7902) |
| 5 | Message security flag | 2 | Unencrypted messages default to 0x0000 |
| 6 | Device ID | 16 | 16-bit ASCII code |

filed explanation：

- **command code**： The highest bit of the command code sent by the MR7901 to the platform is 0, and the command code that platform responds to the MR79601 at the highest position 1.For example, if the command code of device initiates a registration request to the platform is 0x0008, and the command code for the platform response is 0x8008.
- **protocol version**： Identifies the message protocol version used by the sender of the message, and the receiver can process or reject it according to this number.

  The protocol version is represented by two bytes, with the high byte as the major version number and the low byte as the minor version number. The version numbers are all numbers, for example, version 2.1 is expressed as 02 01.

  Note: V2.0 (specially for MR7902)
- **Safety signs**： It is used to identify the relevant security information of the terminal and the packet in the packet header, and the security verification result of the previous packet.
- If the verification result is wrong, the packet has only the header, that is, only the corresponding identification bit of the security identifier in the header of the original packet needs to be changed and sent back (for encryption-related instructions, see

Chapter 13).

For example, the terminal supports encryption, encryption is enabled, encryption is enabled for this message, and the initial key is used for encryption, which is expressed as 1111 0000 0000 0000 (binary), and expressed in hexadecimal: F0 00 00 00

| **Message security identification (16bits)** | |
|---|---|
| **High to low number of bits (binary)** | **Description** |
| **Terminal and message security identification** | |
| 15th bit<br>■□□□ □□□□ □□□□ □□□□ | The terminal supports encrypted identification bits<br>0-not supported<br>1-Support |
| 14th bit<br>□■□□ □□□□ □□□□ □□□□ | Encryption flag is enabled on this terminal<br>0-not enabled<br>1-enable |
| 13th bit<br>□□■□ □□□□ □□□□ □□□□ | Encrypted flag for this message<br>0-plain text<br>1-ciphertext |
| 12th bit<br>□□□■ □□□□ □□□□ □□□□ | Key identification<br>0-Use factory secret key encryption<br>1-Use new key encryption<br>(It is used to inform the platform that the device's local key status is changed to 1, and the factory default is 0) |
| Other reservations | All 0。 |

eg：55 AA 00 22 00 08 00 00 00 01 00 01 E0 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 C7 2A EF 6E 92 EB E4 E9

| **sof(H)** | **sof(L)** | **len(H)** | **len(L)** | **cmd(H)** | **cmd(L)** | **seq (MSB)** | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 22 | 00 | 08 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 01 | 00 | 01 | F0 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **desc_code* (H)** | **desc_code* (L)** | **reg_code* (MSB)** | | | **reg_code* (LSB)** |
| 39 | 36 | 00 | 7D | BC | 56 | CB | 97 | E8 |
| **crc16(H)** | **crc16(L)** | | | | | | | |

| | |
|---|---|
| E4 | E9 |

message header： 00 22 00 08 00 00 00 01 00 01 E0 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00

among them:

length : 0x0022
command code: 0x0008
SN: 0x00000001 Message content encryption uses the first group of key encryption
version : 0.1
security signs : 0xE000， That is, the terminal supports encryption and encryption is enabled. This message is cipher text and is encrypted with the factory key
device id : “861694034205896”

## 6.3Message contentMessage content

The message content, according to different command codes, contains different message bodies. For the specific format, please refer to the command description in the back (Chapter 7).

# 7.Data interaction

Specifically introduces the message format of different commands.

note:

For messages sent by the device to the server, the server must respond accordingly, otherwise the device will repeatedly send the same message.

## 7.1Registered （0x0008/ 0x8008）

### 7.1.1 Command frame definition

Initiated by the device to the platform, the command code: 0x0008, platform confirmation code: 0x8008.

The device sends a registration message to the server (load balancing server)

After the server receives, respond to the registration status with the load server's IP and port.

**note**:

If the device registration is unsuccessful, the registration message will continue to be sent.

The service content of the device registration includes a 2-byte device type description and a 4-byte registration code. The registration code is calculated by the device ID through a fixed algorithm.The specific algorithm is defined separately by the system.

**command code**： 0x0008

**message contentmessage content**：As shown in the table below

| Data segment | Bytes | Description |
|---|---|---|
| Device | 2 | The high byte is the device type |

| | | |
|---|---|---|
| description | | **0x01** ——Data gateway<br>**0x02** —— RFID reader<br>**0x03** —— Computer<br><br>The low byte is the device model code<br><br>**0x01** —— MR7901<br>**0x02** —— MR7901P（safe city/WiFi probe）<br>**0x03** —— MR7902<br>**0x04** —— MR7901K（school attendance）<br>**0x05** —— HX402<br>**0x06** —— MR3202E (assets management)<br>**0x07** —— HT101<br>**0x08** —— MR7901K-1 （ integrated reader for E-bike directional identify）<br>**0x09** —— MR7901P-1（read MR1054x tag）<br>**0x10** —— **HX102（temperature measurement reader）**<br><br>… |
| Registration code | 4 | |

eg: 55 AA 00 22 00 08 00 00 00 01 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 02 78 56 34 12 7E 32

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 22 | 00 | 08 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 01 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **desc_code (H)** | **desc_code (L)** | **reg_code (MSB)** | | | **reg_code (LSB)** |
| 39 | 36 | 00 | 01 | 02 | 78 | 56 | 32 | 12 |
| **crc16(H)** | **crc16(L)** | | | | | | | |
| 7E | 32 | | | | | | | |

Start flag

Start flag sof : 0x55AA

Message header

message length len : 0x0022

command code cmd : 0x0008

Message serial number seq : 0x00000001

协议版本protocol version : 0x0001 (V0.1)

pro_ver

security flag seq_flag : 0x8000 (This terminal supports encryption, encryption is not turned on (clear text transmission), and the local storage is the factory secret key)(This terminal supports encryption, encryption is not turned on (clear text transmission), and the local storage is the factory secret key)

device ID dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in stringin string: "861694034205896")

**message content**

Device description desc_code : 01 02 （ Device type: 01 Internet of Things gateway, model type: 02 is MR7901P）

Registration code reg_code : 78 56 34 12

**check**

crc16 : 0x7E32

## 7.1.2 Platform validation package definitionPlatform validation package definition

Platform validation service content includes the registration result and the current real time, and the load server IP and port assigned to the device login.

Validation code： 0x8008

**message contentmessage content：**

| Data segment | Bytes | Description |
|---|---|---|
| Registration results | 1 | 0x00 — — Registered successfully (if the registration is successful, return to the load server IP and port)<br>0xFE ——Registration code error<br>0xFF ——Registration refused |
| Real time | 6 | Year Month Day Hours Minutes Seconds, year is 2000 |
| IP Load server IP | 32 | String type<br>egg： "218.17.157.214" |
| Load the server port | 2 | Unsigned integer<br>Low byte first, high byte in the post |

eg： 55 AA 00 45 80 08 00 00 00 01 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 12 0B 0D 09 1C 23 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24 13 75 A6

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|

| 55 | AA | 00 | 45 | 80 | 08 | 00 | 00 | 00 |
|---|---|---|---|---|---|---|---|---|
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev id(MSB)** | | | |
| 01 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev id(LSB)** | **reg_status** | **time （MSB）** | | | | |
| 39 | 36 | 00 | 00 | 12 | 0B | 0D | 09 | 1C |
| **time （LSB）** | **IP(MSB)** | **…** | **IP(LSB)** | **port(L)** | **port(H)** | **crc16(H)** | **crc16(L)** | |
| 23 | 32 | … | 00 | 24 | 13 | 75 | A6 | |

**start flagstart flag**

sof : 0x55AA

**message header**

len : 0x0045

cmd : 0x8008

seq : 0x00000001

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000 (No encryption, clear text transmission)

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in stringin string: “861694034205896”)

**message contentmessage content**

reg_status : 00 registration success

time : 12 0B 0D 09 1C 23 The current time of the platform, corresponding to the year, month, day, hour, minute, second, the starting time is 2000

Load server IP : 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 (“218.17.157.214”)

Load server port : 24 13 that is 0x1324 （4900 in DEC）(Note: low byte first)

**Check**

crc16 : 0x75A6

## 7.2Logon （0x0001/ 0x8001）

Initiated by the device to the platform，command code:0x0001，platform validation code: 0x8001。

After the device is successfully registered, it will send a login message to the load server. After a successful login, it will send a heartbeat or other data to the server (load server).

**Note：**

。If the device fails to log in (without receiving the correct response), it will send 10 logon request messages (command code0x0001) repeatedly. After 10 times, the device will send a registration message again to the load balancing server.

## Command package definitionCommand package definition

。Login service content includes the software version number of the device and the CRC16 check of the device's configuration parameter list.

**command code：** 0x0001

**message contentmessage content：**

| Data segment | Bytes | Description |
|---|---|---|
| software version | 2 | main-version number and minor-version number |
| CRC16 check of the device's configuration parameter | 2 | The CRC16 check of the configuration parameter list. The server may decide whether or not to update the parameters accordingly.(High byte first) |

eg: 55 AA 00 20 00 01 00 00 00 00 02 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 05 02 69 E4 23

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 20 | 00 | 01 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 02 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (MSL)** | **ver (H)** | **ver (L)** | **parm_crc 16(H)** | **parm_crc 16(L)** | **crc16 (H)** | **crc16 (L)** |
| 39 | 36 | 00 | 01 | 05 | 02 | 69 | E4 | 23 |

**start flagstart flog**

sof : 0x55AA

**message headermessage header**

len : 0x0020

cmd : 0x0001

seq : 0x00000002
pro_ver : 0x0001 (V0.1)
seq_flag : 0x8000 (This terminal supports encryption, encryption is not turned on (clear text transmission), and the local storage is the factory secret key)
dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message contentmessage content**

ver : 01 05 （Device software version V1.5）
crc16 configuration : 02 69

**check**

crc16 : 0xE423

## 7.2.1 Platform validation package definitionPlatform validation package definition

The platform validation service content includes the login result (1 byte) and the current real time

Validation code: 0x8001

Service content:

| Data segment | Bytes | Description |
|---|---|---|
| Log in results | 1 | 0x00 ——Login successful, no operation request<br>0x02 ——The login is successful and requires the firmware to be updated<br>0x03 ——Login is successful, request to upload device hardware information<br>0x10 ——The login is successful and requires updating the user configuration parameters<br>0xFE ——Login error<br>0xFF ——Login refused (the device received a deny login message, 3 minutes later to send a registration message) |
| Real time | 6 | Year Month Day Hours Minutes Seconds, year 2000 |

eg: 55 AA 00 23 80 01 00 00 00 02 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 12 0B 0D 09 1C 30 81 CB

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 23 | 80 | 01 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |

| 00 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **login_status** | **time （MSB）** | | | | |
| 39 | 36 | 00 | 00 | 12 | 0B | 0D | 09 | 1C |
| **time （LSB）** | **crc16 (H)** | **crc16 (L)** | | | | | | |
| 30 | 81 | CB | | | | | | |

**start flag**

sof : 0x55AA

**message header**

len : 0x0023

cmd : 0x8001

seq : 0x00000000

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

login_status : 00 （success）

time : 12 0B 0D 09 1C 30 corresponding year, month, day, hour, minute, second, the starting time is 2000（09:28:48 November 13, 2018）

**check**

crc16 : 0x81CB

The validation package of requests device update

eg2: 55 AA 00 23 80 01 00 00 00 02 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 02 12 0B 0D 09 1C 30 E1 28

## 7.3Heartbeat （0x0003/ 0x8003）

Initiated by the device to the platform, command code:0x0003, plat validation code: 0x8003。

Note:

If the device does not receive the correct response message after sending the heartbeat message, it will send 5 times the heartbeat message (command code0x0003). After 5 times, the device will send the login message (command code0x0001) again to the load service.

### 7.3.1 Command package definitionCommand package definition

The heartbeat service content includes the working status of the device (2 bytes) and the current state of the device.

command code: 0x0003

Service content:

| Data segment | Bytes | Description |
|---|---|---|
| Device working status | 2 | FORMAT SPECIFICATION:<br>Low 4 byte, connection::<br>The 0bit : GPRS connection, 1 effective<br>The 1bit: LAN connection, 1 effective<br>The 2bit: reserved<br>The 3bit: reserved<br><br>The 4bit: Tag transmission flag , 0: transfer tag records to platform; 1: Do not transfer tag records to the platform<br>The 5bit: Device power failure flag, 为 0: device normal power supply, 1: device external power supply is disconnected<br>The 6～7bit: reserved<br><br>The 8～11bit: battery voltage, 0～10, respectively, there are 0 to 100% of the electricity. (This function will be achieved in MR7901 V3.2 or above version, the corresponding relationship between battery and battery voltage refer to Chapter 11)<br>The 12～15bit: reserved<br>eg:<br>00 01 ——GPRS connection, reported tag data<br>00 11 ——GPRS connection, do not report tag data<br><br>08 03 ——80%GPRS connection, wired connection, report tag data, battery power 80% |
| Device status | 2 | (E-bike V2.9,V3.0 and above)<br>0～7bit : gprs signal strength, the normal range: 0～31, if value is 99, the access signal strength failure |

| | | |
|---|---|---|
| | | 8～11bit: Packet transmission mode (only one bit valid)<br>8bit : Send packets through gprs<br>9bit : Send packets over a wired network<br>10bit : reserved<br>11bit : reserved<br><br>12～15bit: reserved<br><br>eg:<br>11 01 ——gprs signal strength 17,send this heartbeat packet via gprs |
| Device version | 2 | |
| Device time | 6 | Year, month, day, hour, minute, second |

eg: 55 AA 00 28 00 03 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 01 10 01 01 05 12 0B 0D 09 1C 31 6D EF

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 28 | 00 | 03 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **work_state （H）** | **work_state （L）** | **dev_state （H）** | **dev_state （L）** | **time （MSB）** | |
| 39 | 36 | 00 | 00 | 01 | 01 | 05 | 12 | 0B |
| | | | **time （LSB）** | **crc16 (H)** | **crc16 (L)** | | | |
| 0D | 09 | 1C | 31 | 6D | EF | | | |

**start flagstart flag**

sof : 0x55AA

**message header**

len : 0x0028

cmd : 0x0003

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000 (This terminal supports encryption, encryption is not turned on (clear text transmission), and the local storage is the factory secret key)

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: “861694034205896”)

**message content**

work_status : 00 01

gprs connection

Upload tag data

External power supply is normal

The battery power is 0 (MR7901P does not have a battery, this function is not supported)

device status : 10 01

0x10 : gprs RSSI is 16;

0x01 : gprs transfer

software ver : 01 05 (software version V1.5)

time : 12 0B 0D 09 1C 31 corresponding year, month, day, hour, minute, second, the starting time is 2000 (09:28:49 November 13, 2018,)

**check**

crc16 : 0x6DEF

## 7.3.2 Platform validation package definitionPlatform validation package definition

Platform validation service content includes the operation indication (1 byte) and the current real time time (6 bytes, year, month, hour, minute, year 2000)

validation code: 0x8003

Service content:

| Data segment | Bytes | Description |
|---|---|---|
| Operation instructions | 1 | 0x00 ——There is no operational indication<br>0x02 ——Requires updating firmware<br>0x03 ——Reset device (device received, do not respond, restart directly)<br>0x04 ——Update the antenna firmware<br>0x05 ——Get antenna information (version, gain and RSSI)<br>0x06 ——Set device time (after the device is received, set the time, do not respond)<br>**0x07** —— Update reporting tag identification<br>0x08 ——Clear tag data buffer (after the device is received, clear tag data buffer, do not respond)<br>**0x09** ——update IP2<br>**0x0A** —— Request to update encrypted transmission ID (control whether encryption is enabled)<br>**0x0B** —— Request to update the encryption key (encryption must be enabled before the device responds) |

| | | |
|---|---|---|
| | | 0x10 ——Request to update the user configuration parameters<br>0x11 ——Get device status<br>0x12 ——Request to upload device hardware information<br>**0x20** —— Request to send a message to the tag<br>**0x21** —— Number of tag messages to be uploaded<br>**0x22** —— Request to clear the tag message to be sent |
| Real time | 6 | year, month, hour, minute, second. year is 2000 (Platform current time) |

eg: 55 AA 00 23 80 03 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 12 0B 0D 09 1C 31 07 F2

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 23 | 80 | 03 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **ask** | **time （MSB）** | | | | |
| 39 | 36 | 00 | 00 | 12 | 0B | 0D | 09 | 1C |
| **time （LSB）** | **crc16 (H)** | **crc16 (L)** | | | | | | |
| 31 | 07 | F2 | | | | | | |

**start flag**

sof : 0x55AA

**message header**

len : 0x0023

cmd : 0x8003

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000 (No encryption, clear text transmission)

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

ask : 00 No operation

time : 12 0B 0D 09 1C 31 Respectively, the year, month, day, hour, minute, second, the starting time is 2000 (09:28:49 November 13, 2018,)

**check**

crc16 : 0x07F2

## 7.4Data reporting （0x0004/ 0x8004）

Initiated by the device to the platform, command code:0x0004, plat validation code: 0x8004。

Note:

If the device sends a data message but does not receive the correct response message from the platform, it will send 5 times data messages (command code0x0004)

After 5 times,it will send login message to load server （command code0x0001）

### 7.4.1 Command package definitionCommand package definition

The data reporting service content includes several data TLVs

command code: 0x0004

Service content:

| Data segment | Bytes | Description |
|---|---|---|
| TLV | 2+2+XX | Tag type (2Bytes), tag data length (2Btyes), tag data (length reference Chapter 8 TLV index)<br>Defined by the specific data content, the specific format see Chapter 8 (tag format description) |
| TLV | 2+2+XX | |
| …… | | |

TLV structure is as follows:

| TLV type (2 bytes) | LENGTH(2 bytes) | VALUE(The length is defined by LENGTH) |
|---|---|---|

eg: 55 AA 00 46 00 04 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 8B 01 00 11 01 20 78 2B 6A A4 2F 00 00 00 A9 12 0B 0D 09 1C 31 8B 01 00 11 01 20 EB 14 4A 33 64 00 00 00 B8 12 0B 0D 09 1C 31 81 4C

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 46 | 00 | 04 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **TLV (MSB)** | | **…** | **TLV (MSL)** | **crc16 (H)** | **crc16 (L)** |
| 39 | 36 | 00 | 8B | 01 | … | 31 | 83 | 3F |

**start flag**

start flag sof : 0x55AA

**message header**

len : 0x0046

cmd : 0x0004

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000 (This terminal supports encryption, encryption is not turned on (clear text transmission), and the local storage is the factory secret key)

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

TLV data : 8B 01 00 11 01 20 78 2B 6A A4 2F 00 00 00 A9 12 0B 0D 09 1C 31 8B 01 00 11 01 20 EB 14 4A 33 64 00 00 00 B8 12 0B 0D 09 1C 31 (There are 2 tag records. For the specific format, please see the tag description in Chapter 8)

one TLV is: 8B 01 00 11 01 20 78 2B 6A A4 2F 00 00 00 A9 12 0B 0D 09 1C 31 ( Analyze as follows:

TLV type : 0x8B01 RFID tag

TLV data length : 0x0011

TLV data : 01 20 78 2B 6A A4 2F 00 00 00 A9 12 0B 0D 09 1C 31 For the format, please refer to the description of tag data format in Section 8.2)

**Check**

crc16 : 0x814C

## 7.4.2 Platform validation package definition

validation code: 0x8004

Service content: same as heartbeat service content。

# 7.5 Firmware update (0x000D/ 0x800D)

The device sends a firmware download request to the platform, command code: 0x000D, platform validation code: 0x800D.

When the platform needs to send new firmware (such as host firmware, antenna firmware) to the device, fill in the firmware update operation instruction code in the login package, heartbeat packet, and packet confirmation package. After the device receives the update operation instruction, the device requests the firmware update to the platform with command code0x000D.

## 7.5.1 Command package definition

firmware update includes update host firmware and antenna firmware,service content definition as follows:

**command code:** 0x000D

Service content:

| Data segment | Bytes | Description |
|---|---|---|
| file_type | 1 | **0x01** ——host firmware<br>**0x02** ——antenna firmware<br>Other files are not yet defined |
| data_type | 1 | mask flag:<br>**0x00** ——Application file information, including file version, file name, crc16 check, size;<br>**0x01** ——Application file content,when this bit is 1, the following two data segments (block size / block serial number) is valid;<br>**0x02** ——The upgrade is complete (followed by 3 bytes, where the first byte is the upgrade status, 0x01: the upgrade is successful, 0x00: the upgrade failed; the second and third bytes are the upgraded version information). |
| Block size | 2 | This section is valid when the application category is "0x01 - Request File Content"Each packet carries the file block size,Unit: Byte |
| Block index | 2 | This section is valid when the application category is "0x01 - Request File Content"<br>The file block number applied to the platform |

1, Application file information

When the device receives the upgrade instruction code, it sends the request message of the upgrade file information to the server, that is, the application file information.

Service content:

| Data segment | Bytes | Description |
|---|---|---|
| file_type | 1 | 0x01 ——host firmware<br>0x02 ——antenna firmware<br>Other files are not yet defined |
| data_type | 1 | mask flag:<br>0x00 ——Application file information, including file version, file name, crc16 check, size; |
| Reserved | 4 | 00 00 00 00 |

**eg1:** 55 AA 00 22 00 0D 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 00 00 00 00 00 56 48

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 22 | 00 | 0D | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **file_type** | **data_type** | **Reserved (MSB)** | | | **Reserved (LSB)** |
| 39 | 36 | 00 | 01 | 00 | 00 | 00 | 00 | 00 |
| **crc16 (H)** | **crc16 (L)** | | | | | | | |
| 56 | 48 | | | | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0022

cmd : 0x000D

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

Request upgrade file type : 01 (update host firmware)

**file_type**

Request data type : 00 (Request upgrade file information)

**data_type**

**Reserved** : 00 00 00 00 （reserved）

**check**

crc16 : 0x5648

2, Application file content

After receiving the correct upgrade file information from the platform, the device sends the "application file content" message to the platform.

**Service content:**

| Data segment | Bytes | Description |
|---|---|---|
| file_type | 1 | 0x01 —— host firmware<br>0x02 —— antenna firmware |
| data_type | 1 | 0x01 —— Application file content |
| block_size | 2 | Transfer block size of update file. (High byte first)<br>The server according to this value, read the size of the block_size data sent to the device, the last packet is less than block_size, filled with 0.<br>01 DD (477Bytes) |
| block_index | 2 | The block index of file (High byte first) |

**eg2:** 55 AA 00 22 00 0D 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 01 01 DD 00 00 C7 5A

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 22 | 00 | 0D | 00 | 00 | 00 |
| seq (LSB) | pro_ver (H) | pro_ver (L) | sec_flag (H) | sec_flag (L) | dev_id (MSB) | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | dev_id | file_type | data_typ | block_siz | block_ | block_in | block_in |

| | | (LSB) | | e | e(H) | size(L) | dex(H) | dex(L) |
|---|---|---|---|---|---|---|---|---|
| 39 | 36 | 00 | 01 | 01 | 01 | DD | 00 | 00 |
| crc16 (H) | crc16 (L) | | | | | | | |
| C7 | 5A | | | | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0022

cmd : 0x000D

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

**file_type** : 01 (update host firmware)

**data_type** : 01 (Request to upgrade the contents of the file)

**block_size** : 0x01DD (send block size of update file content 0x1DDBytes)

**block_index** : 0x0000 (update block 0 on file)

**check**

crc16 : 0xC75A

**1. The upgrade is complete**

After host and antenna update file transfer are completed, will send update successful message to server.

| Data segment | Bytes | Description |
|---|---|---|
| file_type | 1 | 0x01 ——host firmware<br>0x02 ——antenna firmware |
| data_type | 1 | 0x02 ——update completed |
| update_flag | 1 | Used to mark whether the upgrade file is received correctly by the device<br>00 : Upgrade failed<br>01 : update successed |
| update_ver | 2 | The firmware version after the upgrade<br>eg:<br>01 05 : updated firmware is V1.5 |
| reserved | 1 | |

**eg3 :** 55 AA 00 22 00 0D 00 00 00 5E 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36

00 01 02 01 01 05 00 08 10

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 22 | 00 | 0D | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 5E | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **file_type** | **data_type** | **update_flag** | **update_ver (MSB)** | **update_ver (LSB)** | **Reserved** |
| 39 | 36 | 00 | 01 | 02 | 01 | 01 | 05 | 00 |
| **crc16 (H)** | **crc16 (L)** | | | | | | | |
| 08 | 10 | | | | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0022

cmd : 0x000D

seq : 0x0000005E

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**file_type** : 01 (Upgrade host firmware(update host firmware)

**data_type** : 02 (update completed)

**update_flag** : 01 (Receive the upgrade file successfully)

**update_ver** : 01 05 ( The version of the upgrade file received is V1.5)

**Reserved** : 00 (reserved）

**check**

crc16 : 0x0810

## 7.5.2 Platform validation package definition

Platform validation service content based on content application

validation code: 0x800D

Service content:

| Data segment | Bytes | Description |
|---|---|---|
| data_type | 1 | **0x01** ——Host file basic information<br><br>**0x02** ——Host file content data<br>**0x03** ——Antenna file basic information<br>**0x04** ——Antenna file content data<br>**0x05** ——The upgrade file ends |
| File basic information | 16 | When the data type is 0x01 / 0x03, this data segment exists<br>5 bytes file name, 3 byte extension, 2 byte file version, 2 byte file crc16 check (reserved), 4 byte file size |
| data length | 2 | |
| data | | |

**1,** Response to application file information

1. When the platform receives the "Request File Message" sent from the device, the information of the upgrade file is sent to the device.

**2. Service content:**

| Data segment | Bytes | Description |
|---|---|---|
| data_type | 1 | 0x01 ——Host firmware file basic information<br>0x03 — — Antenna firmware file basic information |
| file_name | 5 | |
| file_ext | 3 | |
| file_version | 2 | |
| file_crc16 | 2 | The overall CRC16 checksum for the bit upgrade file(See Chapter 11 for the calibration algorithm)High byte first |
| file_size | 4 | High byte first |

**eg1:** 55 AA 00 2D 80 0D 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 00 00 00 00 00 00 00 00 00 00 B5 87 00 01 0A EE 1A 8E

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 2D | 80 | 0D | 00 | 00 | 00 |

| seq (LSB) | pro_ver (H) | pro_ver (L) | sec_flag (H) | sec_flag (L) | dev_id (MSB) | | | |
|---|---|---|---|---|---|---|---|---|
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | dev_id (LSB) | data_type | file_name | file_ext | file_version | file_crc16 | file_size |
| 39 | 36 | 00 | 01 | 00 00 00 00 00 | 00 00 00 | 00 00 | B5 87 | 00 01 0A EE |
| crc16 (H) | crc16 (L) | | | | | | | |
| 1A | 8E | | | | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x002D

cmd : 0x800D

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**file_type** : 01 (update host firmware)

**file_name** : 00 00 00 00 00 (5Byes, update file name)

**file_ext** : 00 00 00 (3Byes, Upgrade the file extension)

**file_version** : 00 00 (2Byes, Upgrade the file version)

**file_crc16** : B5 87 (2Byes, Upgrade file crc16 check is 0xB587)

**file_size** : 00 01 0A EE (4Byes, Upgrade file size is 0x00010AEE，High byte first)

**check**

crc16 : 0x1A8E

2, response application file content

When the platform receives the "Request File Content" message from the device, the content of the upgrade file is issued according to the application file block size.

| Data segment | Bytes | Description |
|---|---|---|
| data_type | 1 | 0x02 ——Host firmware file content data<br>0x04 ——Antenna firmware file content data |
| block_size | 2 | This package contains the block size of the upgrade file data. High byte first<br>01 DD (477Bytes), at this time the frame length of the |

| | | |
|---|---|---|
| | | entire communication frame is 512Bytes |
| data<br>update_data | block_size | update file data<br>The data length is determined by block_size |

**eg2：** 55 AA 01 FC 80 0D 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 02 01 DD F0 17 … 11 5C 56

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 01 | FC | 80 | 0D | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **data_type** | **block_size(H)** | **block_size(L)** | **update_data（MSB）** | **…** | **update_data (LSB)** |
| 39 | 36 | 00 | 01 | 01 | DD | F0 | … | 11 |
| **crc16 (H)** | **crc16 (L)** | | | | | | | |
| 5C | 56 | | | | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x021F

cmd : 0x800D

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**file_type** : 01 (Upgrade host firmware)

**block_size** : 01 DD (2Byes,The upgrade file package size is 0x01DD)

**update_data** : F0 … 11 (Upgrade file content)

**check**

crc16 : 0x5C56

3, Respond to upgrade complete

After the device receives the upgrade file, it will send the upgrade complete message to the platform,The platform receives a confirmation message.

| Data segment | Bytes | Description |
|---|---|---|
| data_type | 1 | 0x05——The upgrade file ends |

**eg3：** 55 AA 00 1D 80 0D 00 00 00 5E 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 05 7C 66

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1D | 80 | 0D | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 5E | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **data_type** | **crc16 (H)** | **crc16 (L)** | | | |
| 39 | 36 | 00 | 05 | 7C | 66 | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001D

cmd : 0x800D

seq : 0x0000005E

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**file_type** : 05 (After the upgrade is confirmed, the device will immediately reset after receiving this message, and run according to the new firmware to start)

**check**

crc16 : 0x7C66

## 7.6Configuration parameters （0x000A/ 0x800A）

Initiated by the device to the platform，command code:0x000A，platform validation code：0x800A。

When the platform needs to acquire or configure the device parameters, the operation instructions are filled in the login packet, heartbeat packet, and packet response. After receiving the operation instruction code, the equipment sends the corresponding parameter information to the platform.

### 7.6.1 Command package definition 0x000A

Service content includes parameter type and parameters

command code: 0x000A

Service content:

| SN | Data segment | Bytes | Description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x00** —— Report system parameters<br>**0x01** —— )Report IP2's ip and port (this message is only valid under IP2 connection)<br>**0x02** —— report antenna info (version, gain,rssi)<br><br>**0x03** —— Report transmission tag data identification<br>**0x04** —— Report encrypted sign<br>**0x05** —— Report encryption key<br><br>**0x10** —— report user parameters<br>**0x11** —— report device statues<br>**0x12** —— report device hardware info<br>**0x80** —— Report configuration confirmation message |
| 2 | parameters | x | **When the first byte is 0x00:**<br>240Bytes, system configuration parameters, please refer to Chapter 9 for configuration parameter format description<br>**When the first byte is 0x02:**<br>12Bytes, corresponding to the version of 4 antennas, gain, rssi<br>**When the first byte is 0x03:**<br>1Byte, 1 does not report tag data, 0 reports tag data<br>**When the first byte is 0x04:**<br>1Byte, 1 uses encrypted message transmission, 0 does not use encrypted message transmission<br>**When the first byte is 0x05:**<br>256 Byte, 16 sets of keys<br><br>**When the first byte is 0x10:**<br>182Bytes, user configuration parameters, please refer to |

| | | | |
|---|---|---|---|
| | | | Chapter 9 Configuration Parameter Format Description for specific format description<br>**When the first byte is 0x11：**<br>106Byte, the current status information of the device<br>**When the first byte is 0x12：**<br>127Byte, device hardware information<br>**When the first byte is 0x80：**<br>1Byte, 1 configuration parameter success, 0: configuration failure |

**1. Report system parameters 0x00**

The device reports or updates the system parameters according to the operation instruction code carried in the confirmation message issued by the platform (During login, heartbeat or data transfer). Report the system parameters of the device according to this message

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x00 —— Report system parameters** |
| 2 | parameters | 240 | For the specific format, please refer to Chapter 9, System Configuration Parameter Description |

eg1： 55 AA 01 0D 00 0A 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 55 01 01 05 01 00 B4 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 00 01 C0 A8 01 C7 FF FF FF 00 C0 A8 01 01 64 00 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24 13 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F8 11 00 2E 12 3C 00 25 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 02 00 02 00 FF FF 10 4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 A8 A8 A8 A8 1F 1F 1F 1F 00 A1 00 05 00 00 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 C0 12 A5 8E 28 00 00 02 00 0F 47 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 4A FF

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 01 | 0D | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm ...** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 00 | ... | 4A | FF | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x010D

cmd : 0x000A

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

**parm_type**

00 : 0. (**Report system parameters**) 31st byte in the frame (counting from 0)

**parm** : (For specific format description, please refer to Chapter 9 Configuration Parameter Format Description)

55 : 1. The 32nd byte in the frame, the parameter characteristic value, when reading the configuration parameter, it is fixed at 0x55

01 : 2. The 33rd byte in the frame, the working mode is:

1>. GPRS（Lower four definitions： 0x01: GPRS 0x02: LAN）

2>. Transfer tag records to the platform

3>. Unencrypted (message content is not encrypted, transmitted in clear text)

01 05 : 3. Starting from the 34th byte in the frame, firmware version V1.5 (main version number 1, from version 5)

01 : 4. 36th byte in the frame, buzzer logo, turn on the buzzer

（ 0x00: off，

0x01： on）

00 : 5. The 37th byte in the frame, GPRS module type: 0x00: SIM800C

B4 00 : 6. starting at the 38th byte in the frame, the tag deduplication filtering time is 0x00B4, which is 180 seconds (low byte first, high byte second), which is the time to judge the tag leaving the base station

38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 : 7. Start with the 40th byte in the frame, device ID, "861694034205896"

00 00 : 8. Starting at the 56th byte in the frame, the base station stays regularly reported function is disabled (V3.3 version)

01 : 9. The 58th byte in the frame, DHCP is enabled (0x00: off, 0x01: on, for LAN)

C0 A8 01 C7 : 10. Start at the 59th byte in the frame, LAN local IP, 192.168.1.199

FF FF FF 00 : 11.

Starting at the 63rd byte in the frame, the LAN subnet mask is 255.255.255.0

C0 A8 01 01 : 12.

Starting at the 67th byte in the frame, the LAN gateway 192.168.1.1

64 00 : 13.

Starting at the 71st byte in the frame, the local LAN IP port is 0x0064, which is 100 (low byte first, high byte second)

32 31 38 2E 31 37 2E 31 35 : 14.

37 2E 32 31 34 00 00 00

thresholds for antennas 1, 2, 3, and 4 are -88dBm, -88dBm, -88dBm, -88dBm

1F 1F 1F 1F : 24.

Beginning at the 204th byte in the frame, the gains of antennas 1, 2, 3, and 4 are 31dBm, 31dBm, 31dBm, and 31dBm, respectively

00 : 25.

The 208th byte in the frame, the Bluetooth output tag identification, 0x00 does not output the tag, 0x01 outputs the tag

A1 : 26.

The 209th byte in the frame, the communication connection status, 0xA1: the device has established a communication connection with the platform through GPRS (if it is 0xA2: the device and the platform have established a communication connection through the LAN, 0xA3: the device and the platform communicate through GPRS and LAN connection)

00 : 27.

210th byte in the frame, tag filter identification, 0x00: no filter is used

05 : 28.

The 211th byte in the frame, the tag filtering interval is 0x05, which is 5 seconds

00 00 : 29. Start at the 212th byte in the frame, reserved 2

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 30. Starting with the 214th byte in the frame, the IP address of IP2 is "218.17.157.214"

5C 12 : 31 Start with the 246th byte in the frame, port 0x125C of IP2 (low byte first)

A5 : 32 Starting from the 248th byte in the frame, the antenna firmware upgrade indicator has a valid upgrade file (valid upgrade file is stored in the device)

8E 28 00 00 : 33 Starting from the 249th byte in the frame, the file size of the antenna firmware upgrade, 0x0000288E (low byte first)

02 00 : 34 Starting from the 253rd byte in the frame, the antenna firmware upgrade file version V2.00

0F 47 : 35 Starting from the 255th byte in the frame, the crc16 check of the antenna firmware upgrade file (low byte first)

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 36 Start at the 257th byte in the frame, reserve 3

**check**

crc16 : 0x4AFF

**2. Report IP2 IP and port 0x01**

This message is only valid when the device is connected to IP2 (generally to modify IP2, use this message to obtain IP2 first, and then modify IP2).

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x01 —— Report IP2's ip and port (this message is only valid under IP2 connection)** |
| 2 | IP of IP2 | 32 | String type, IP of IP2 management platform |
| 3 | Port of IP2 | 2 | Low byte first |

eg2：55 AA 00 3F 00 0A 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 23 13 10 C6

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 3F | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm ...** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 01 | ... | 10 | C6 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x003F

cmd : 0x000A

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 01 **(Report IP2 IP and port)**

**parm**

IP address of IP2 : 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 (IP2的IP ”218.17.157.214”)

IP2 port : 23 13 (IP2 port, 0x1323, which is 4899 (low byte first))

**check**

crc16 : 0x10C6

**3.** Report antenna info **0x02**

The antenna information contains the current version of the 4 antennas, gain, and filter the RSSI threshold.

**message content:**

| SN | Data segment | Bytes | Description |
|---|---|---|---|
| 1 | param_type | 1 | 0x02 ——report antenna info |
| 2 | ant_fw_ver | 8 | 1 to 4 antenna version information, each antenna version in 2 bytes<br>If it is FF FF, it means that the antenna version information of the channel is not read |
| 3 | ant_gain | 4 | 1 ~ 4 antenna gain (range 0 ~ 31) If it is FF, it means that the antenna gain of the channel is not read |
| 4 | ant_rssi | 4 | 1 ~ 4 antenna filter threshold (range -1 ~ -128)<br>If it is 01, it means that the RSSI of the channel is not read |

eg3: 55 AA 00 2D 00 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 02 02 00 02 00 02 00 FF FF 1F 1F 1F FF A8 A8 A8 01 AC DE

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 2D | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **ant1_fw_ver(H)** | **ant1_fw_ver(L)** | **ant2_fw_ver(H)** | **ant2_fw_ver(L)** | **ant3_fw_ver(H)** |
| 39 | 36 | 00 | 02 | 02 | 00 | 02 | 00 | 02 |
| **ant3_fw_ver(L)** | **ant4_fw_ver(H)** | **ant4_fw_ver(L)** | **ant1_gain** | **ant2_gain** | **ant3_gain** | **ant4_gain** | **ant1_rssi** | **ant2_rssi** |
| **00** | **FF** | **FF** | **1F** | **1F** | **1F** | **FF** | **A8** | *A8* |
| **ant3_rssi** | **ant4_rssi** | **crc16 (H)** | **crc16 (L)** | | | | | |
| **A8** | **01** | **AC** | **DE** | | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x002D

cmd : 0x000A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 02 (**report** antenna info)

**ant_version** : 02 00 02 00 02 00 FF FF1 to 4 antenna version information, each antenna version in 2 bytes If it is FF FF, it means that the antenna version information of the channel is not read)

explain:

antenna 1 firmware; V2.0

antenna 2 firmware; V2.0

antenna 3 firmware; V2.0

antenna 4 did not get the firmware version information.

**ant_gain** : 1F 1F 1F FF （1 ~ 4 antenna gain (range 0 ~ 31) FF, invalid)

explain:

antenna 1 gain: 31

antenna 2 gain: 31

antenna 3 gain: 31

antenna 4 gain: not get

**ant_rssi** : A8 A8 A8 01 1 ~ 4 antenna filter threshold (range -1 ~ -128) 0 is not filter.1 means that the RSSI of the channel is not read)

explain:

antenna 1 rssi: -88

antenna 2 gain: -88

antenna 3 gain: -88

antenna 4 gain: not get

**check**

crc16 : 0xACDE

**4. Report transmission tag data identification 0x03**

The transmission tag data identification is used to control whether the device reports the tag data to the platform. The factory default is to report to the platform. If it is set to not report, the device will not report to the platform after reading the number of tags.

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x03 —— Report transmission tag data identification** |
| 2 | Transmission tag data identification | 1 | 00 Report tag data to the platform<br>01 Does not report tag data to the platform |

eg4: 55 AA 00 1E 00 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 03 00 43 74

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1E | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 03 | 00 | 43 | 74 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E

cmd : 0x000A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 03 (**Report transmission tag data identification**)

**parm** : 00 (Report tag data to the platform)

**Value description**:

1. 0x00Report tag data to the platform
2. 0x01Does not report tag data to the platform

**check**

crc16 : 0x4374

**5. Report encryption flag 0x04**

The encryption flag is used to control whether the device uses encrypted ciphertext transmission. The factory default is unencrypted, that is, plaintext transmission. If it is set to encryption, the data packets uploaded by the device to the platform are transmitted using encryption (for encryption-related instructions, see Chapter 13).

**message content:**

| Sn | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x04** —— Report encryption flag |
| 2 | Transmission tag data identification | 1 | 00 The message content is not encrypted, and it is transmitted in clear text<br>01 Encrypt message content, use ciphertext transmission |

eg5: 55 AA 00 1E 00 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 04 00 DA E3

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1E | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 04 | 00 | DA | E3 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E

cmd : 0x000A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000 This terminal supports encryption, the local is the factory secret key

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 04 (Report encryption flag)

**parm** : 00 (none-encrypted message content)

**Value description**:

1. 0x00 The message content is not encrypted, and it is transmitted in clear text

2. 0x01 Encrypt message content, use ciphertext transmission

**check**

crc16 : 0xDAE3

**6. Report the encryption key 0x05 (ciphertext transmission)**

Report the secret keys used for the encryption algorithm, a total of 16 groups, each group of 16 bytes (for the AES algorithm, using CFB mode, 128-bit secret key) (for encryption related instructions, please see Chapter 13).

**note:**

When you use the encryption key, you must first enable the encryption function, that is, use ciphertext transmission, and pass the secret key to the server.

**message content:**

| sn | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x05** —— Report the encryption key (clear text is 0x05) |
| 2 | Transmission tag data identification | 256 | 16 sets of secret keys, each group of 16 bytes |

eg6: 55 AA 01 1D 00 0A 00 00 00 07 00 01 E0 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 0D F6 5B 52 EB A7 D4 A9 B2 22 E5 E1 B0 0F 03 53 55 F4 28 C9 E2 09 1F 10 42 02 D9 0F BC E8 AA AF FA F5 1F B9 1B 4A 9A A0 51 45 54 BC 59 25 91 1E 71 83 B2 45 5E 6C 6A B7 C3 D2 4D DC 07 65 9A CD 3D D9 22 BF 86 22 E4 98 3E 55 81 5B 12 7A D6 E9 03 CB 81 10 50 E8 E8 A7 38 D4 57 FA 6B C7 C5 E6 46 43 66 6F CD BA 50 99 6E 6D 7E 78 F3 D9 88 A6 1F 22 5F B8 F7 15 46 D5 2E 49 38 0F AA C8 54 8F 75 08 90 02 75 27 8C F3 09 3F 33 4B BC 49 F9 8A ED 5F 3D A4 36 18 1C 55 FE F3 C1 5B D0 16 6A 90 05 8C 86 97 00 0C E0 57 F1 DB F9 57 EC 93 D6 72 77 51 EA E6 A2 81 5E B1 51 48 E6 38 2C F8 37 6A DF 73 F9 13 9D FB 67 E1 E4 73 3C C0 AC 15 36 06 28 ED C6 18 C7 4B 21 17 BB 85 28 46 D9 C2 02 40 C9 8B B6 21 2D 92 AF 03 38 74 CB 9A 14 54 C6 4C D4 02 85 6F 04 FC CE C7 19 60 D4 DE 10 F0 49 B5 73 ED 85

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 01 | 1D | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 07 | 00 | 01 | F0 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |

| | | dev_id (LSB) | parm_type* | key[0]* | … | key[255]* | crc16 (H) | crc16 (L) |
|---|---|---|---|---|---|---|---|---|
| 39 | 36 | 00 | 03 | F6 | … | 73 | ED | 85 |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E

cmd : 0x000A

seq : 0x00000007 Message content encryption uses the seventh group of key encryption

pro_ver : 0x0001 (V0.1)

seq_flag : 0xE000 The terminal supports encryption, encryption is enabled, this message is cipher text, and the factory key group is used for encryption

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 0D It is a ciphertext, which is 0x05 after decryption, that is, the report key

**parm** : F6 5B …B5 73

It is the factory secret key group (a total of 16 groups, each group of 16 bytes) (for cipher text, according to the serial number and security identification, we know that we need to use the seventh group of factory secret key decryption), after decryption: A1 B0… BE AD

**check**

crc16 : 0xED85

**7.** Report user configuration parameters **0x10**

**message content:**

| Item | Data segment | The No. of bytes | Description |
|---|---|---|---|
| 1 | Parameter type param_type | 1 | 0x10 —— Report user configuration parameters |
| 2 | Parameter | 182 | Take Chapter 9 for the reference. |

eg7: 55 AA 00 D3 00 0A 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 10 55 01 01 05 01 00 B4 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 00 01 C0 A8 01 C7 FF FF FF 00 C0 A8 01 01 64 00 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24 13 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00

00 00 00 00 00 F8 11 00 2E 12 3C 00 25 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 02 00 02 00 FF FF 10 4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 A8 A8 A8 A8 1F 1F 1F 1F 00 A1 00 05 00 00 16 9D

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | D3 | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm ...** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 10 | ... | 16 | 9D | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x00D3

cmd : 0x000A

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

**parm_type**

10 : 0. （Report user configuration parameters），31st byte in the frame (counting from 0)

**parm** : (For specific format description, please refer to Chapter 9 Configuration Parameter Format Description)

55 : 1. The 32nd byte in the frame, the parameter characteristic value, when reading the configuration parameter, it is fixed at 0x55

01 : 2. The 33rd byte in the frame, the working mode is:

1>. GPRS（Lower four definitions： 0x01: GPRS 0x02: LAN）

2>. Transfer tag records to the platform

3>. Unencrypted (message content is not encrypted, transmitted in clear text)

01 05 : 3. Starting from the 34th byte in the frame, firmware version V1.5 (main version number 1, from version 5)

01 : 4. The 36th byte in the frame, the buzzer logo, turn on the buzzer (0x00: closed,

0x01: open)

00 : 5.The 37th byte in the frame, GPRS module type: 0x00: SIM800C

B4 00 : 6. Starting from the 38th byte in the frame, the tag deduplication filtering time is 0x00B4, which is 180 seconds (low byte first, high byte second), which is the time to judge the tag leaving the base station

38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 : 7. Start with the 40th byte in the frame, device ID, "861694034205896"

00 00 : 8. Beginning at the 56th byte in the frame, the base station stays regularly reported function is disabled (V3.3 version)

01 : 9. The 58th byte in the frame, DHCP is enabled (0x00: off, 0x01: on, for LAN)

C0 A8 01 C7 : 10. Start at the 59th byte in the frame, LAN local IP, 192.168.1.199

FF FF FF 00 : 11. Starting at the 63rd byte in the frame, the LAN subnet mask is 255.255.255.0

C0 A8 01 01 : 12. Starting at the 67th byte in the frame, the LAN gateway 192.168.1.1

64 00 : 13. Starting at the 71st byte in the frame, the local LAN IP port is 0x0064, which is 100 (low byte first, high byte second)

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 14. Starting from the 73rd byte in the frame, the IP of GPRS server 1, "218.17.157.214", platform IP (balanced server address)

F8 11 : 15. Starting from the 105th byte in the frame, the port of GPRS server 1, 0x11F8 is 4600 (low byte first, high byte second), platform port

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 16. Starting at the 107th byte in the frame, the IP of LAN server 1, "218.17.157.214", platform IP (balanced server address)

F8 11 : 17. Starting at the 139th byte in the frame, the port of LAN server 1, 0x11F8, which is 4600 (low byte first, high byte second)

00 2E 12 3C 00 25 : 18. Starting from the 141th byte in the frame, the local MAC address of the LAN is 00-2E-12-3C-00-25

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 19. Start at the 147th byte in the frame, reserved 1

02 00 02 00 02 00 FF FF : 20. Starting from the 175th byte in the frame, the antenna firmware version corresponds to the firmware version of antennas 1 to 4, each antenna version occupies 2 bytes. When it is FF FF, it means that reading the antenna version failed. That is, the firmware versions of antennas 1 to 4 are: V2.0, V2.0, V2.0, none (Failed to read antenna 4, the antenna may not be connected to channel 4)

10 : 21. 183th byte in the frame, GPRS signal strength 0x10(16)

4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 : 22. Starting with the 184th byte in the frame, the device number is "MR7901P-03C0025"

A8 A8 A8 A8 : 23. Starting with the 200th byte in the frame, the rssi filtering thresholds for antennas 1, 2, 3, and 4 are -88dBm, -88dBm, -88dBm, -88dBm

1F 1F 1F 1F : 24. Beginning at the 204th byte in the frame, the gains of antennas 1, 2, 3, and 4 are 31dBm, 31dBm, 31dBm, and 31dBm, respectively

00 : 25. The 208th byte in the frame, the Bluetooth output tag identification, 0x00 does not output the tag, 0x01 outputs the tag

A1 : 26.The 209th byte in the frame, the communication connection status, 0xA1: the device has established a communication connection with the platform through GPRS (if it is 0xA2: the device and the platform have established a communication connection through the LAN, 0xA3: the device and the platform communicate through GPRS and LAN connection)

00 : 27. 210th byte in the frame, tag filter identification, 0x00: no filter is used

05 : 28.The 211th byte in the frame, the tag filtering interval is 0x05, which is 5 seconds

00 00 : 29. Start at the 212th byte in the frame, reserved 2

**check**

crc16 : 0x169D

**8.** Report equipment status **0x11**

Used to obtain the current state of the device

| Item | Data segment | Byte | Description |
|---|---|---|---|
| 1 | param_type | 1 | **0x11** —— eport equipment status |
| 2 | fw_version | 2 | eg:<br>03 00 —— Host firmware version V3.0 |
| 3 | gprs_ip | 32 | **String type**<br>eg:<br>“218.17.157.214” |
| 4 | gprs_port | 2 | Low byte ahead |
| 5 | gprs_link_status | 1 | **0x00** —— No connection to the platform<br>**0x01** —— The platform has connections |
| 6 | lan_ip | 32 | **String type** |
| 7 | lan_port | 2 | Low byte ahead |
| 8 | lan_link_status | 1 | **0x00** —— No connection to the platform |

| | | | |
|---|---|---|---|
| | | | 0x01 —— The platform has connections |
| 9 | gprs_bffer_cnt | 2 | The number of tags sent to the platform via GPRS<br>Low byte ahead |
| 10 | lan_bffer_cnt | 2 | Number of tags sent to the platform via wire<br>Low byte ahead |
| 11 | tag_filt_cnt | 2 | The number of tags used to filter or pre-judge<br>Low byte ahead |
| 12 | ant_fw_version | 8 | The version information of 1 ~ 4 antenna, the version information of each antenna occupies 2 bytes<br>The version information that is not read to the antenna is not read for FF FF |
| 13 | ant_gain | 4 | 1 ~ 4 antenna gain Settings<br>Value range: 0～31<br>In the case of FF, the antenna gain is not obtained |
| 14 | ant_rssi | 4 | Rssi threshold setting for antenna 1 ~ 4 antenna<br>Range of values: -1 ~ 128 (band symbol single-byte, singed char)<br>At 1, the rssi is not obtained |
| 15 | gprs_flash_tag_cnt | 4 | Stored in flash, the number of tags to be sent to the platform via GPRS<br>Low byte ahead |
| 16 | lan_flash_tag_cnt | 4 | Stored in flash, ready to be sent to the platform by wire.<br>Low byte ahead |
| 17 | flash_log_cnt | 4 | Low byte ahead |

**message content:**

eg8: 55 AA 00 87 00 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 11 01 05 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24 13 01 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 25 13 00 18 00 18 00 08 00 02 00 02 00 02 00 FF FF 1F 1F 1F FF A8 A8 A8 01 00 00 00 00 00 00 00 00 74 00 00 00 CE 88

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 87 | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |

| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
|---|---|---|---|---|---|---|---|---|
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 11 | ... | CE | 88 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0087

cmd : 0x000A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 11 1. (Report equipment status)

Current device host firmware version : 01 05 2. (host firmware V1.5)

Current gprs connection IP : 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 3. (Current gprs connection IP，“218.17.157.214”)

Current gprs connection IP port : 24 13 4. (The IP port of the current gprs connection, 0x1324, which is 4900, with the low byte first)

Current gprs connection status : 01 5. (The current device has established a connection with the platform through gprs)

Current wired connection IP : 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 6. (The current device is wired to the IP, "218.17.157.214")

Current wired IP port : 25 13 7. (The current wired IP port, 0x1325, or 4901, with the low byte first)

Current wired connection status : 00 8. (The current device is connected to the platform via cable, and no connection is established)

Number of gprs tag sending buffers : 18 00 9. (Number of tags sent to the platform through gprs in the cache 0x0018, low byte first)

Number of wired tag sending buffers : 18 00 10. (Number of tags sent to the platform by wire in the cache 0x0018, low byte first)

Filter or determine the number of tags : 08 00 11. (Number of tags sent to the platform by wire in the cache 0x0018, low byte first)

Antenna current firmware version 02 00 02 00 02 00 FF FF 12. (1~4 antenna version information, among which

The version of antenna 1 is V2.0

The version of antenna 2 is V2.0
The version of antenna 3 is V2.0
(No version is read for antenna 4)

Antenna current gain setting : 1F 1F 1F FF 13. (1~4 antenna gain setting, including:
Antenna 1 does not read gain
Antenna 2 does not read gain
The gain of antenna 3 is 1
No gain is read from antenna 4)

Antenna current RSSI threshold setting : A8 A8 A8 01 14.(1~4 antenna rssi threshold setting, among them:
The rssi of antenna 1 is set to 0xA8, which is -88dBm
The rssi of antenna 2 is set to 0xA8, which is -88dBm
The rssi of antenna 3 is set to 0xA8, which is -88dBm
No. 4 antenna does not read rssi value
(Is a signed single byte number, single char)

Number of gprs tags in flash : 00 00 00 00 15. (In flash, the number of tags sent to the platform through gprs is 0x000000, with the low byte first)

Number of wired tags in flash : 00 00 00 00 16. (In flash, the number of tags sent to the platform by wire is 0x000000, with the low byte first)

Number of log records in flash : 74 00 00 00 17. (In flash, the number of tags sent to the platform by wire is 0x000074, with the low byte first)

**check**

crc16 : 0x CE88

**9. Report equipment hardware information 0x12**

Equipment hardware information, current communication equipment ID, hardware ID, product number, GPRS module IMEI, SIM card CCID, battery power, external power identification, etc

Report: 128 bytes, the following table

| Item | Data segment | Bytes | Descritption |
|---|---|---|---|
| 1 | param_type | 1 | 0x12 —— Report device hardware information |
| 2 | cur_device_id | 16 | String, the ID used in the current communication header, the last byte is 0 |
| 3 | mcu_device_id | 16 | The string, the last byte is 0 |
| 4 | product_sn | 16 | The string, the last byte is 0 |
| 5 | gprs_imei | 16 | The string, the last byte is 0 |
| 6 | sim_ccid | 21 | The string, the last byte is 0 |
| 7 | battery_level | 1 | Value: 0 ~ 10, 10 represents 100% electricity |
| 8 | power_link_status | 1 | 0: external power supply normal, 1: external power supply disconnect |
| 9 | reserved | 40 | Reserved |

eg9： 55 AA 00 9C 00 0A 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 12 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 34 33 35 35 31 30 35 30 30 33 43 30 30 32 35 00 4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 38 39 38 36 30 32 62 31 31 39 31 36 39 30 30 30 32 33 32 39 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 11 BE

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 01 | 9C | 00 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm ...** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 12 | ... | 11 | BE | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x019C

cmd : 0x000A

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 12 1 .(Report device hardware information) 31st byte in frame (counting from 0)

**cur_device_id** : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 2. “861694034205896”(Start at the 32nd byte in the frame)

**mcu_device_id** : 34 33 35 35 31 30 35 30 30 33 43 30 30 32 35 00 3. “4355105003C0025”（Start at the 48th byte in the frame）

**product_sn** : 4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 4. “MR7901P-03C0025”（Start at the 64th byte in the frame）

**gprs_imei** : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 5. “861694034205896”（Start at the 80th byte in the frame）

**sim_ccid** : 38 39 38 36 30 32 62 31 31 39 31 36 39 30 30 30 32 33 32 39 00 6. “898602b1191690002329”（Start at the 96th byte in the frame）

**battery_level** : 00 7. Battery level 00 (MR7901P has no battery) (starting at the 117th byte in the frame)

**power_link_status** : 00 8. Connected to an external power supply (starting at the 118th byte in the frame)

**reserved** : 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 9. Reserved (starting at the 119th byte in the frame)

**check**

crc16 : 0x11BE

**10.** Confirmation of information **0x80**

When the configuration message of the platform is received, such as configuring the user configuration parameter, configuring the antenna parameter, and so on, the device will respond to the following message to the platform, responding to the result of the configuration parameters of the platform.

Note: after the device responds to this message, it will be restarted immediately and run with the correct parameters configured.

**message content:**

| Item | Data segment | bytes | Description |
|---|---|---|---|
| 1 | param_type | 1 | 0x80 —— Report configuration confirmation message (V2.8 support) |
| 2 | return_opt | 1 | 0x01 —— Configuration parameter success<br>0x00 —— Configuration parameter failed |

eg10: 55 AA 00 1E 00 0A 00 00 00 04 00 01 80 00 34 33 35 35 31 30 33 30 30 33 45 30 30 33 39 00 80 01 E7 69

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1E | 00 | 0A | 00 | 00 | 00 |
| seq (LSB) | pro_ver (H) | pro_ver (L) | sec_flag (H) | sec_flag (L) | dev_id (MSB) | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | dev_id (LSB) | parm_type | return_opt | crc16 (H) | crc16 (L) | | |
| 39 | 36 | 00 | 80 | 01 | E7 | 69 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E

cmd : 0x000A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 80 **Report configuration confirmation message (used to respond to the device platform has received the configuration parameters)**

**return_opt** : 01 Configuration parameters successful

**check**

crc16 : 0xE769

## 7.6.2 Platform validation package definition 0x800A

After the platform receives the message of configuration parameters sent on the device (the command code is 0x000A), configure the configuration message, such as the configuration parameter system parameter, configuration antenna parameter, and so on, as needed.

Service Content: Contains parameter types and parameters

Confirmation code: 0x800A

**message content:**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | type | 1 | 0x00-configure system parameters<br>0x01-configure IP2 ip and port<br>0x02-configure antenna gain, rssi<br>0x03 -Configure whether to report tag data<br>0x04 -Configure whether to encrypt the message<br>0x05-Configure encryption key<br><br>0x10-configure user parameters<br>0x80-platform confirmation message |
| 2 | parameters | x | **When the first byte is 0x00:**<br>**If it is the entire parameter 240Bytes (see Chapter 9)**<br>**When the first byte is 0x01:** |

| | | | |
|---|---|---|---|
| | | | **32Bytes IP2 + 2Bytes port (low byte first)**<br>**When the first byte is 0x02:**<br>**8Bytes antenna gain and RSSI**<br>**When the first byte is 0x03:**<br>**1Byte, 0x01 does not report the tag to the platform; 0x00 reports the tag to the platform**<br>**When the first byte is 0x04:**<br>**1Byte, 0x01 encrypted message; 0x00 unencrypted message**<br>**When the first byte is 0x05:**<br>**256Byte, 16 sets of secret keys**<br><br>**When the first byte is 0x10:**<br>**182Bytes, user configuration parameters**<br>**When the first byte is 0x80:**<br>**1Byte, 1: The platform successfully received the configuration parameters; 0:** |
| | | | |

**1. Configure system parameters 0x00**

After the platform receives the configuration message (command code 0x000A) sent from the device, if you need to modify the device parameters, you can issue commands to modify the system parameters

**message content:**

| SN | datafile | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x00 : Configure system parameters (see Chapter 9)** |
| 2 | parameters | 240 | For the specific format, please refer to Chapter 9, System Configuration Parameter Description |

**note:**

After receiving this message, the device will restart immediately.

eg1: 55 AA 01 0D 80 0A 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 55 01 01 05 01 00 B4 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 00 01 C0 A8 01 C7 FF FF FF 00 C0 A8 01 01 64 00 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24 13 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 F8 11 00 2E 12 3C 00 25 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 02 00 02 00 02 00 FF FF 10 4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 A8 A8 A8 A8 1F 1F 1F 1F 00 A1 00 05 00 00 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 C0 12 A5 8E 28 00 00 02 00 0F 47 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 70 6B

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq | | |
|---|---|---|---|---|---|---|---|---|

| | | | | | | (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 01 | 0D | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm …** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 00 | … | 70 | 6B | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x010D

cmd : 0x800A

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

**parm_type**

00 : 0.(Configure system parameters) 31st byte in the frame (counting from 0)

**parm** : (For specific format description, please refer to Chapter 9 Configuration Parameter Format Description)

55 : 1. The 32nd byte in the frame, the parameter characteristic value, when reading the configuration parameter, it is fixed at 0x55

01 : 2. The 33rd byte in the frame, the working mode is: GPRS transmission, and the tag record is transmitted to the platform (the lower four bits are defined: 0x01: GPRS 0x02: LAN)

01 05 : 3. Starting from the 34th byte in the frame, firmware version V1.5 (main version number 1, from version 5)

01 : 4. The 36th byte in the frame, buzzer identification, turn on the buzzer (0x00: off, 0x01: on)

00 : 5. The 37th byte in the frame, GPRS module type: 0x00: SIM800C

B4 00 : 6. Starting from the 38th byte in the frame, the tag deduplication filtering time is 0x00B4, which is 180 seconds (low byte first, high byte second), which is the time to judge the tag leaving the base station

38 36 31 36 39 34 30 33 34 : 7. Start with the 40th byte in the frame, device ID,

32 30 35 38 39 36 00 "861694034205896"

00 00 : 8. Beginning at the 56th byte in the frame, the base station stays regularly reported function is disabled (V3.3 version)

01 : 9. The 58th byte in the frame, DHCP is enabled (0x00: off, 0x01: on, for LAN)

C0 A8 01 C7 : 10. Start at the 59th byte in the frame, LAN local IP, 192.168.1.199

FF FF FF 00 : 11. Starting at the 63rd byte in the frame, the LAN subnet mask is 255.255.255.0

C0 A8 01 01 : 12. Starting at the 67th byte in the frame, the LAN gateway 192.168.1.1

64 00 : 13. Starting at the 71st byte in the frame, the local LAN IP port is 0x0064, which is 100 (low byte first, high byte second)

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 14. Starting from the 73rd byte in the frame, the IP of GPRS server 1, "218.17.157.214", platform IP (balanced server address)

F8 11 : 15. Starting from the 105th byte in the frame, the port of GPRS server 1, 0x11F8 is 4600 (low byte first, high byte second), platform port

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 16. Starting at the 107th byte in the frame, the IP of LAN server 1, "218.17.157.214", platform IP (balanced server address)

F8 11 : 17. Starting at the 139th byte in the frame, the port of LAN server 1, 0x11F8, which is 4600 (low byte first, high byte second)

00 2E 12 3C 00 25 : 18. Starting from the 141th byte in the frame, the local MAC address of the LAN is 00-2E-12-3C-00-25

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 19. Start at the 147th byte in the frame, reserved 1

02 00 02 00 02 00 FF FF : 20. Starting from the 175th byte in the frame, the antenna firmware version corresponds to the firmware version of antennas 1 to 4, each antenna version occupies 2 bytes. When it is FF FF, it means that reading the antenna version failed. That is, the firmware versions of antennas 1 to 4 are: V2.0, V2.0, V2.0, none (Failed to read antenna 4, the antenna may not be connected to channel 4)

10 : 21. 183th byte in the frame, GPRS signal strength 0x10(16)

4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 : 22. Starting with the 184th byte in the frame, the device number is "MR7901P-03C0025"

A8 A8 A8 A8 : 23. Starting with the 200th byte in the frame, the rssi filtering thresholds for antennas 1, 2, 3, and 4 are -88dBm, -88dBm, -88dBm, -88dBm

1F 1F 1F 1F : 24. Beginning at the 204th byte in the frame, the gains of antennas 1, 2, 3, and 4 are 31dBm, 31dBm, 31dBm, and 31dBm, respectively

00 : 25. The 208th byte in the frame, the Bluetooth output tag identification, 0x00 does not output the tag, 0x01 outputs the tag

A1 : 26. The 209th byte in the frame, the communication connection status, 0xA1: the device has established a communication connection with the platform through GPRS (if it is 0xA2: the device and the platform have established a communication connection through the LAN, 0xA3: the device and the platform communicate through GPRS and LAN connection)

00 : 27. 210th byte in the frame, tag filter identification, 0x00: no filter is used

05 : 28. The 211th byte in the frame, the tag filtering interval is 0x05, which is 5 seconds

00 00 : 29. Start at the 212th byte in the frame, reserved 2

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 30. Starting with the 214th byte in the frame, the IP address of IP2 is "218.17.157.214"

5C 12 : 31 Start with the 246th byte in the frame, port 0x125C of IP2 (low byte first)

A5 : 32 Starting from the 248th byte in the frame, the antenna firmware upgrade indicator has a valid upgrade file (valid upgrade file is stored in the device)

8E 28 00 00 : 33 Starting from the 249th byte in the frame, the file size of the antenna firmware upgrade, 0x0000288E (low byte first)

02 00 : 34 Starting from the 253rd byte in the frame, the antenna firmware upgrade file version V2.00

0F 47 : 35 Starting from the 255th byte in the frame, the crc16 check of the antenna firmware upgrade file (low byte first)

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 36 Start at the 257th byte in the frame, reserve 3

**check**

crc16 : 0x706B

**2. Configure IP2 IP and port 0x01**

**message content:**

| SN | Data file | Bytes | Description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x01 —— Configure IP2 ip and port (this message can only be valid under IP2 connection)** |
| 2 | IP2's ip | 32 | String type, IP of IP2 management platform |

| | | | |
|---|---|---|---|
| 3 | IP2 port | 2 | Low byte first |

**note:**

**This message is only valid when the device is connected to IP2, and after receiving the correct message, the device will restart.**

eg2： 55 AA 00 3F 80 0A 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 C0 12 78 E8

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 3F | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm ...** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 01 | ... | 78 | E8 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x003F

cmd : 0x800A

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 01 **(Configure IP2 ip and port)**

**parm**

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : IP2 IP address(IP2 IP ”218.17.157.214”)

C0 12 : P2 port(Low byte first) (0x12C0，that is 4800)

**check**

crc16 : 0x78E8

### 3. Configure antenna parameters 0x02

It is used to configure the gain of the antenna and filter the RSSI threshold.

**message content：**

| Item | Data | bytes | Description |
|---|---|---|---|

| | segment | | |
|---|---|---|---|
| 1 | param_type | 1 | 0x02 —— **Configure antenna parameters** |
| 2 | ant_gain | 4 | Gain of antenna 1 ~ 4 (range 0 ~ 31)<br>If it is FF, it indicates that the antenna gain is not read to the channel |
| 3 | ant_rssi | 4 | 1 ~ 4 antenna filter threshold value (range -1 ~ 128)<br>If it is 01, it indicates that the antenna is not read to the channel RSSI |

eg3： 55 AA 00 25 80 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 02 1F 1F 1F 1F A8 A8 A8 A8 E0 80

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 25 | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **ant1_gain** | **ant2_gain** | **ant3_gain** | **ant4_gain** | **ant1_rssi** |
| 39 | 36 | 00 | 02 | 1F | 1F | 1F | 1F | *A8* |
| **ant2_rssi** | **ant3_rssi** | **ant4_rssi** | **crc16 (H)** | **crc16 (L)** | | | | |
| A8 | A8 | A8 | E0 | 80 | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0025

cmd : 0x800A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 02 (**Configure antenna parameters**)

**ant_gain** : 1F 1F 1F FF (Respectively corresponding to the gain of antenna 1~4, value range 0~31)

Resolution:

Antenna 1 gain: 31

Antenna 2 gain: 31

Antenna 3 gain: 31
Antenna 4 gain: 31

**ant_rssi** : A8 A8 A8 A8 (Respectively corresponding to the filtering threshold of the signal strength of the antennas 1 to 4, the value range: 0 to -128)

Resolution:
Rssi for antenna 1: -88
Rssi for antenna 2: -88
Rssi for antenna 3: -88
Rssi for antenna 3: -88

**check**

crc16 : 0xE080

**4. Config Transmission tag data identification 0x03**

Used to configure whether the device reports tag data to the platform.

**message content:**

| Item | Data segment | bytes | Description |
|---|---|---|---|
| 1 | param_type | 1 | **0x03 ——config Transmission tag data identification** |
| 2 | Tag no. | 1 | 00 Report the tag data to the platform<br>01 Do not report the tag data to the platform |

eg4: 55 AA 00 1E 80 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 03 01 2B 26

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1E | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 03 | 01 | 2B | 26 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E
cmd : 0x800A
seq : 0x00000004
pro_ver : 0x0001 (V0.1)
seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 03 (config **Transmission tag data identification**)

**parm** : 01 : Does not report tag data to the platform

**check**

crc16 : 0x2B26

**5. Configure encryption ID 0x04**

The platform sends the new encrypted ID to the device to enable or disable the encryption function.

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x04** —— Report encrypted ID |
| 2 | Transmission tag data identification | 1 | 00 The message content is not encrypted, and it is transmitted in clear text<br>01 Encrypt message content, use ciphertext transmission |

eg5: 55 AA 00 1E 80 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 04 01 B2 B1

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1E | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 04 | 01 | B2 | B1 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E

cmd : 0x800A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000 This terminal supports encryption, the local is the factory secret key

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 04 (**Configure encryption ID**)

**parm** : 01 (Turn on message content encryption)

**Value description**:

1. 0x00 The message content is not encrypted, and it is transmitted in clear text

2. 0x01 Encrypt message content, use ciphertext transmission

**check**

crc16 : 0xB2B1

**6. Configure encryption key 0x05 （Ciphertext transmission）**

Report the secret keys used for the encryption algorithm, a total of 16 groups, each group of 16 bytes (for the AES algorithm, using CFB mode, 128-bit secret key).

**Note: When you upload the encryption key, you must first enable the encryption function, that is, use ciphertext transmission, and pass the secret key to the server.**

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x05** —— Report the encryption key (clear text is 0x05) |
| 2 | Transmission tag data identification | 256 | 16 sets of secret keys, each group of 16 bytes |

eg6: 55 AA 01 1D 80 0A 00 00 00 07 00 01 E0 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 0D FD 2A 2D A5 F9 B8 E5 A3 73 88 84 8C 3B 39 20 4D 49 E7 AC 42 81 5A 51 56 0C 10 2A E9 81 2D D5 A3 8C 56 42 C5 3E DD C1 95 62 AE 46 B5 5E 0A A0 78 2D 93 ED 89 A4 30 C2 BF 89 A3 A6 D6 54 80 72 1F D0 3C 16 1E E4 3E CB 20 4E 33 EC 64 3D 70 A5 95 A5 8B 7D 97 18 06 BB 65 47 40 8D 5C A0 03 F4 85 75 D9 3A F0 06 E6 91 A5 DC 9B 7C 00 42 6C FE 8A CC D1 B1 FD B3 01 EA 75 EE 84 E7 E5 0F D5 C0 82 9D B6 FB 39 D2 E9 F4 E3 D6 00 01 96 7E 59 FE DE 23 9F DC E5 3F EA 86 4F FE 45 70 B2 27 67 23 93 97 B5 E6 F8 A2 6D 6A 30 CB 06 F5 98 4A 22 82 74 B0 52 6F ED 5C E7 C8 CF C9 6D 85 A2 88 3F 98 DC 5C E2 AA 87 59 05 D5 BB 3B D1 C1 B7 4E 93 7B 4A F2 3F 64 3B E9 7E DC 67 F5 BD 2B 8D B0 F1 8A 1D 01 5A 59 A8 8E 98 5E 38 9C A8 DC C3 A7 E2 C1 AB 5E B5 9D DC 05 85 8C 21 87 43 8E 87 DB FE 3F FF 78 A2

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 01 | 1D | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 07 | 00 | 01 | E0 | 00 | 38 | 36 | 31 | 36 |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type*** | **key[0]*** | **...** | **key[255]*** | **crc16 (H)** | **crc16 (L)** |
| 39 | 36 | 00 | 0D | FD | ... | FF | 78 | A2 |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E

cmd : 0x000A

seq : 0x00000007 Message content encryption uses the seventh group of key encryption

pro_ver : 0x0001 (V0.1)

seq_flag : 0xE000 The terminal supports encryption, encryption is enabled, this message is cipher text, and the factory key group is used for encryption

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 0D It is a ciphertext, which is 0x05 after decryption, that is, to modify the secret key

**parm** : FD 2A …3F FF New key group (a total of 16 groups, each group of 16 bytes) (for cipher text, according to the serial number and security identification, it is learned that the seventh group of keys must be decrypted from the factory), after decryption: AA C1… 65 29

**check**

crc16 : 0x78A2

**7.** Config user configuration parameters **0x10**

Used to configure some user parameters.

**message content:**

| Item | Data segment | bytes | Description |
|---|---|---|---|
| 1 | param_type | 1 | 0x10 —— Config user configuration parameters |
| 2 | Parameter | 182 | For a specific format, see chapter 9, configuration parameters |

Note: After receiving the correct message, the device will restart.

eg5: 55 AA 00 D3 80 0A 00 00 00 03 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 10 55 01 01 05 01 01 3C 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 00 01 C0 A8 01 C7 FF FF FF

00 C0 A8 01 02 DC FF 32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 24 13 4D 54 49 49 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 23 6D 74 69 32 36 35 34 36 33 39 32 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 14 4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 A8 A8 A8 A8 1F 1F 1F 1F 00 A1 00 05 00 00 AB B9

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | D3 | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **parm ...** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 10 | ... | AB | B9 | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x00D3

cmd : 0x800A

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

**parm_type**

00 : 0.(Set user parameters) 31st byte in the frame (counting from 0)

**parm** : (For specific format description, please refer to Chapter 9 Configuration Parameter Format Description)

55 : 1. The 32nd byte in the frame, the parameter characteristic value, when reading the configuration parameter, it is fixed at 0x55

01 : 2. The 33rd byte in the frame, the working mode is: GPRS transmission, and the tag record is transmitted to the platform (the lower four bits are defined: 0x01: GPRS 0x02: LAN)

01 05 : 3. Starting from the 34th byte in the frame, firmware version V1.5 (main version number 1, from version 5)

01 : 4. The 36th byte in the frame, buzzer identification, turn on the buzzer (0x00: off, 0x01: on)

00 : 5. The 37th byte in the frame, GPRS module type: 0x00: SIM800C

B4 00 : 6. Starting from the 38th byte in the frame, the tag deduplication filtering time is 0x00B4, which is 180 seconds (low byte first, high byte second), which is the time to judge the tag leaving the base station

38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 : 7. Start with the 40th byte in the frame, device ID, "861694034205896"

00 00 : 8. Beginning at the 56th byte in the frame, the base station stays regularly reported function is disabled (V3.3 version)

01 : 9. The 58th byte in the frame, DHCP is enabled (0x00: off, 0x01: on, for LAN)

C0 A8 01 C7 : 10.Start at the 59th byte in the frame, LAN local IP, 192.168.1.199

FF FF FF 00 : 11. Starting at the 63rd byte in the frame, the LAN subnet mask is 255.255.255.0

C0 A8 01 01 : 12. Starting at the 67th byte in the frame, the LAN gateway 192.168.1.1

64 00 : 13. Starting at the 71st byte in the frame, the local LAN IP port is 0x0064, which is 100 (low byte first, high byte second)

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 14. Starting from the 73rd byte in the frame, the IP of GPRS server 1, "218.17.157.214", platform IP (balanced server address)

F8 11 : 15. Starting from the 105th byte in the frame, the port of GPRS server 1, 0x11F8 is 4600 (low byte first, high byte second), platform port

32 31 38 2E 31 37 2E 31 35 37 2E 32 31 34 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 16. Starting at the 107th byte in the frame, the IP of LAN server 1, "218.17.157.214", platform IP (balanced server address)

F8 11 : 17. Starting at the 139th byte in the frame, the port of LAN server 1, 0x11F8, which is 4600 (low byte first, high byte second)

00 2E 12 3C 00 25 : 18. Starting from the 141th byte in the frame, the local MAC address of the LAN is 00-2E-12-3C-00-25

00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 : 19. Start at the 147th byte in the frame, reserved 1

02 00 02 00 02 00 FF FF : 20. Starting from the 175th byte in the frame, the antenna firmware version corresponds to the firmware version of antennas 1 to 4, each antenna version occupies 2 bytes. When it is FF FF, it means that reading the antenna version failed. That is, the firmware versions of antennas 1 to 4 are: V2.0, V2.0, V2.0, none (Failed to read antenna 4, the antenna may not be connected to channel 4)

10 : 21. 183th byte in the frame, GPRS signal strength 0x10(16)

4D 52 37 39 30 31 50 2D 30 33 43 30 30 32 35 00 : 22. Starting with the 184th byte in the frame, the device number is "MR7901P-03C0025"

A8 A8 A8 A8 : 23. Starting with the 200th byte in the frame, the rssi filtering thresholds for antennas 1, 2, 3, and 4 are -88dBm, -88dBm, -88dBm, -88dBm

1F 1F 1F 1F : 24. Beginning at the 204th byte in the frame, the gains of antennas 1, 2, 3, and 4 are 31dBm, 31dBm, 31dBm, and 31dBm, respectively

00 : 25. The 208th byte in the frame, the Bluetooth output tag identification, 0x00 does not output the tag, 0x01 outputs the tag

A1 : 26. The 209th byte in the frame, the communication connection status, 0xA1: the device has established a communication connection with the platform through GPRS (if it is 0xA2: the device and the platform have established a communication connection through the LAN, 0xA3: the device and the platform communicate through GPRS and LAN connection)

00 : 27. 210th byte in the frame, tag filter identification, 0x00: no filter is used

05 : 28. The 211th byte in the frame, the tag filtering interval is 0x05, which is 5 seconds

00 00 : 29. Start at the 212th byte in the frame, reserved 2

**check**

crc16 : 0xABB9

## 8. Platform confirmation message 0x80

The platform receives configuration parameter messages sent by the device, such as reporting system parameters, IP2 parameters, and antenna information. Used to inform the device platform that it has received the reported configuration message.

**message content:**

| Item | Data segment | bytes | Description |
|---|---|---|---|
| 1 | param_type | 1 | 0x80 —— Send platform confirmation message |

eg6: 55 AA 00 1D 80 0A 00 00 00 04 00 01 80 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 80 25 00

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1D | 80 | 0A | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 04 | 00 | 01 | 80 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |

| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
|---|---|---|---|---|---|---|---|---|
| | | **dev_id (LSB)** | **parm_type** | **crc16 (H)** | **crc16 (L)** | | | |
| 39 | 36 | 00 | 80 | 25 | 00 | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001D

cmd : 0x800A

seq : 0x00000004

pro_ver : 0x0001 (V0.1)

seq_flag : 0x8000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 80 **Platform confirmation message (used to respond to the device and inform the device that the platform has received configuration parameters)**

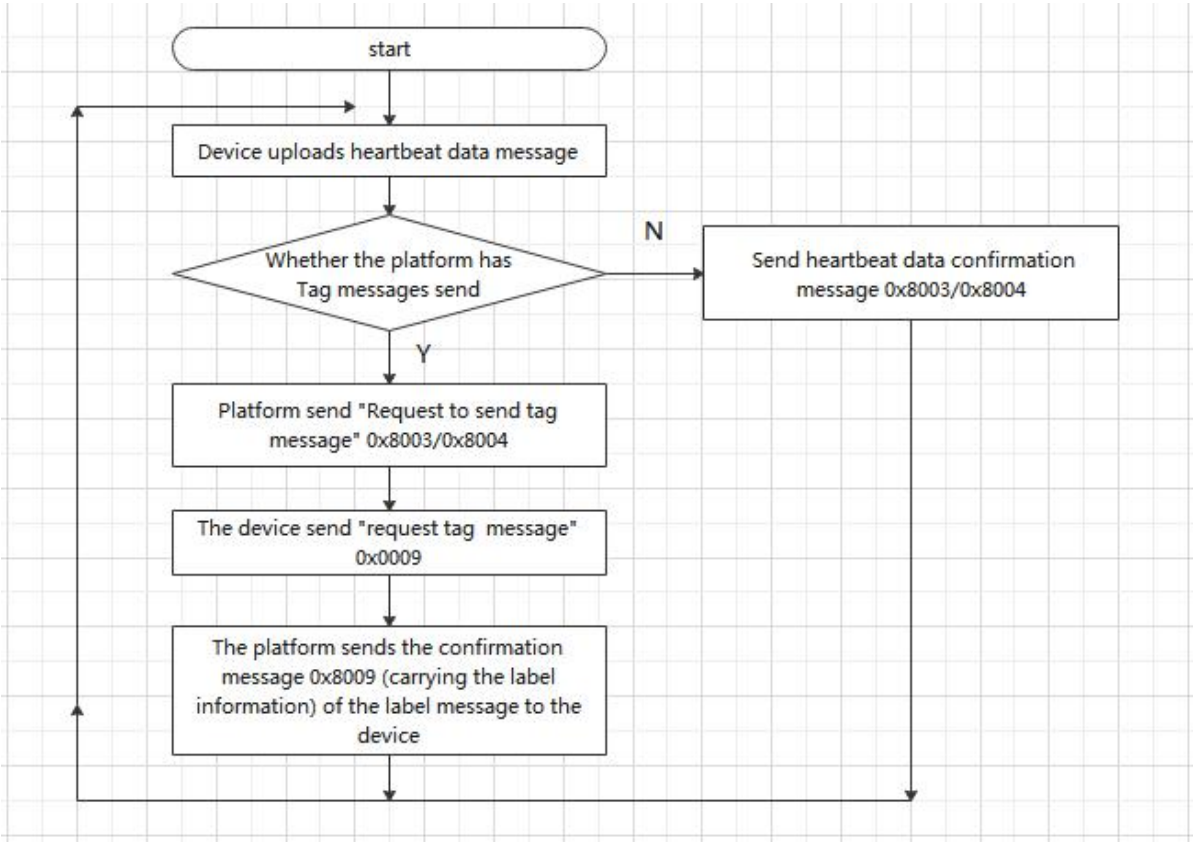
**check**

crc16 : 0x2500

## 7.7Tag massage （0x0009/ 0x8009）

Initiated by the device to the platform, command code: 0x0009, platform confirmation code: 0x8009.

When the platform needs to send a message to the tag, the corresponding operation instruction code needs to be filled in the heartbeat confirmation package (command code 0x8003) and the data upload confirmation package (command code 0x8004). After receiving the operation instruction code, the device sends the corresponding request packet to the platform




Obtain the tag message process from the platform

## 7.7.1 Command package definition 0x0009

The message content contains parameter types and parameters.

**command code：** 0x0009

**message content：**

| SN | Data file | Bytes | Description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x00** —— request tag message<br>**0x01** —— report the tag message processing result<br>**0x02** —— report the number of tag messages to be sent<br>**0x80** —— eport confirmation message |
| 2 | parm | x | **When the first byte is 0x00:**<br>2Bytes, corresponding to the number of tag messages to be sent and the free length of the message queue<br>**When the first byte is 0x01:**<br>7Bytes, a tag message processing result<br>**When the first byte is 0x02:**<br>2Bytes, corresponding to the number of tag messages to be sent and the free length of the message queue |

| | | | |
|---|---|---|---|
| | | | **When the first byte is 0x80:**<br>4Bytes |

**1. Request tag message 0x00**

According to the operation instruction code carried in the confirmation message (in heartbeat or data transmission) issued by the platform-"0x20 request to send a message to the tag", the device sends the "request tag message" to the platform according to this message

**message content:**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | parm_type | 1 | **0x00** —— request tag message |
| 2 | parm | 2 | The first byte is the number of tag messages to be sent by the device<br>The second byte is the idle number of the device tag message queue (you can also store the number of tag messages) |

eg1: 55 AA 00 1F 00 09 00 00 00 03 00 01 00 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 00 32 94 A0

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1F | 00 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (MSL)** | **parm_type** | parm[0] | parm[1] | **crc16 (H)** | **crc16 (L)** | |
| 39 | 36 | 00 | 00 | 00 | 32 | 94 | A0 | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001F

cmd : 0x0009

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 (in string: "861694034205896")

**message content**

**parm_type** : 00 (Request tag message) 31st byte in frame (counting from 0)

**parm**

00 :The 32nd byte in the frame, the number of tag messages to be sent by the device is 0

32 :The 33rd byte in the frame, the number of idle messages in the device tag message queue is 50

**check**

crc16 : 0x94A0

**2. Report the tag message processing results 0x01**

Used by the device to report the results of tag processing.

The tag message is sent from the platform to the device, and finally the message is sent to the antenna. After the antenna is processed, the result of this message will be reported to the platform.

**message content:**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | **parm_type** | 1 | **0x01** —— Report the tag message processing results |
| 2 | Tag_type | 1 | tag type |
| 3 | Tag_id | 4 | tag ID |
| 4 | Msg_opt | 1 | 0x00: Successfully sent (the message has been sent to the tag)<br>0x80: Failed to send (send timeout, message was not sent to the tag) |
| 5 | Msg_sn | 1 | value range0x01～0xFF |

eg2: 55 AA 00 24 00 09 00 00 00 03 00 01 00 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 20 16 10 F4 99 00 01 6C 37

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 2D | 00 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 02 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **Tag_type** | **Tag_id (MSB)** | | | **Tag_id (LSB)** |
| 39 | 36 | 00 | 01 | 20 | 16 | 10 | F4 | 99 |
| **Msg_opt** | **Msg_sn** | **crc16 (H)** | **crc16 (L)** | | | | | |
| 00 | 00 | 6C | 37 | | | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0024

cmd : 0x0009

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 01 (Report the tag message processing results)

Tag_type : 20 (student card )

Tag_id : 16 10 F4 99 (response of Message sent to tag 16 10 F4 99)

Msg_Opt : 00 (Message sent successfully)

Msg_sn : 01 (The message sequence number is 1)

**check**

crc16 : 0x6C37

### 3. Report the number of tag messages to be sent 0x02

It is used to report how many tag messages to be sent in the device.

**message content：**

| SN | date file | bytes | description |
|---|---|---|---|
| 1 | **parm_type** | 1 | **0x02** ——Report the number of tag messages to be sent |
| 2 | **parm** | 2 | The first byte is the number of tag messages to be sent by the device<br>The second byte is the idle number of the device tag message queue (you can also store the number of tag messages) |

eg3: 55 AA 00 1F 00 09 00 00 00 03 00 01 00 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 02 00 32 FA C0

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1F | 00 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |

| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
|---|---|---|---|---|---|---|---|---|
| | | **dev_id (MSL)** | **parm_type** | **parm**[0] | **parm**[1] | **crc16 (H)** | **crc16 (L)** | |
| 39 | 36 | 00 | 02 | 00 | 32 | FA | C0 | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001F

cmd : 0x0009

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 02(Report the number of tag messages to be sent) The 31st byte in the frame (counting from 0)

**parm**

00 :The 32nd byte in the frame, the number of tag messages to be sent by the device is 0

32 :The 33rd byte in the frame, the number of idle messages in the device tag message queue is 50

**check**

crc16 : 0xFAC0

**4. Confirmation message 0x80**

When receiving the platform's tag message, the device will respond to the platform with the following message to inform the platform that the message has been received.

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | **parm_type** | 1 | **0x80** —— Report a confirmation message |
| 2 | **return_opt** | 1 | **0x00 :** Message reception failed<br>**0x01 :** Message received successfully<br>**0x02 :** Message queue is full |
| 3 | **parm** | 2 | The first byte is the number of tag messages to be sent by the device<br>The second byte is the idle number of the device tag message queue (you can also store the number of tag messages) |

eg4: 55 AA 00 20 00 09 00 00 00 06 00 01 00 00 34 33 35 35 31 30 33 30 30 33 45 30 30 33 39 00 80 01 02 30 95 9C

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 20 | 00 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 06 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **return_opt** | **parm**[0] | **parm**[1] | **crc16 (H)** | **crc16 (L)** |
| 39 | 36 | 00 | 80 | 01 | 02 | 30 | 95 | 9C |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0020

cmd : 0x0009

seq : 0x00000006

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 80 Platform confirmation message (used to respond to the device platform has received the tag message)

**return_opt** : 01 Message received successfully

**parm** : 02 30

02 : The number of tag messages to be sent by the device is 2

30 : The number of idle message queues on the device tag is 48

**check**

crc16 : 0x959C

## 7.7.2 Platform validation package definition 0x8009

After receiving the tag message (command code 0x0009) sent by the device, the platform can send the following tag message to the device as needed.

The message content contains parameter types and parameters.

**Confirmation code：** 0x8009

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | type | 1 | **0x00** —— Send tag message<br>**0x01** —— Number of tag messages to be uploaded<br>**0x02** —— Request to clear the tag message to be sent<br>**0x80** —— Platform confirmation message |
| 2 | parameters | x | **When the first byte is 0x00：**<br>40 * x Bytes<br>**When the first byte is 0x01：**<br>none<br>**When the first byte is 0x02：**<br>none<br>**When the first byte is 0x80：**<br>1Byte |

**1. Send tag message 0x00**

After the platform receives the "Request tag Message" (command code 0x0009) sent from the device, if the message needs to be delivered to the tag, this message is sent.

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | **parm_type** | 1 | **0x00 —— Send tag message** |
| 2 | **tag message** TM | x | A tag message TM (see section 7.7.3 for format description) length is not more than 137Bytes<br>Multiple items can be sent together, but the number cannot exceed 7 |

**note:**

**Up to 7 tag messages TM can be sent in one message frame.**

eg1： 55 AA 00 32 80 09 00 00 00 03 00 01 00 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 00 20 16 10 F4 B8 00 01 3C 0C 48 6F 77 20 61 72 65 20 79 6F 75 3F 1E 0E

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 31 | 80 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **TM ...** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 00 | ... | 1E | 0E | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x0031

cmd : 0x8009

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 00 **Send tag message**

**TM** : 20 16 10 F4 B8 00 01 3C 0C 48 6F 77 20 61 72 65 20 79 6F 75 3F

There is only one tag message TM here. For the detailed analysis of TM format, please see section 7.7.3

**check**

crc16 : 0x1E0E

**2. Number of tag messages to be uploaded 0x01**

It is used by the platform to obtain the number of tag messages to be sent in the device.

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | **parm_type** | 1 | **0x01 —— Number of tag messages to be uploaded** |

eg6: 55 AA 00 1D 80 09 00 00 00 03 00 01 00 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 01 C3 CB

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1D | 80 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **crc16 (H)** | **crc16 (L)** | | | |
| 39 | 36 | 00 | 01 | C3 | CB | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001D

cmd : 0x8009

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 01 **Number of tag messages to be uploaded**

**check**

crc16 : 0xC3CB

**3. Request to clear the tag message to be sent 0x02**

Used by the platform to clear all tag messages to be sent

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | **parm_type** | 1 | **0x02 —— Request to clear the tag message to be sent** |

eg6: 55 AA 00 1D 80 09 00 00 00 03 00 01 00 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 02 F3 A8

| **sof(H)** | **sof(L)** | **len(H)** | **len(L)** | **cmd(H)** | **cmd(L)** | **seq (MSB)** | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1D | 80 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **crc16 (H)** | **crc16 (L)** | | | |
| 39 | 36 | 00 | 02 | F3 | A8 | | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001D

cmd : 0x8009

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string:

“861694034205896”）

**message content**

**parm_type** : 02 **Request to clear the tag message to be sent**

**check**

crc16 : 0xF3A8

**4. Platform confirmation message 0x80**

The platform receives the "request tag message" (when no message needs to be sent) sent by the device, "reports the tag processing result" or "reports the number of tag messages to be sent", and sends this message to inform the device that the platform has received Reported news.

**message content：**

| SN | data file | bytes | description |
|---|---|---|---|
| 1 | **parm_type** | 1 | **0x80** —— Send platform confirmation message |
| 2 | **(operation code) opt** | 1 | **0x00 :** Do nothing<br>**0x01 :** Clear the cache to be sent |

eg6: 55 AA 00 1E 80 09 00 00 00 03 00 01 00 00 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 80 00 0B 8D

| sof(H) | sof(L) | len(H) | len(L) | cmd(H) | cmd(L) | seq (MSB) | | |
|---|---|---|---|---|---|---|---|---|
| 55 | AA | 00 | 1D | 80 | 09 | 00 | 00 | 00 |
| **seq (LSB)** | **pro_ver (H)** | **pro_ver (L)** | **sec_flag (H)** | **sec_flag (L)** | **dev_id (MSB)** | | | |
| 03 | 00 | 01 | 00 | 00 | 38 | 36 | 31 | 36 |
| | | | | | | | | |
| 39 | 34 | 30 | 33 | 34 | 32 | 33 | 35 | 38 |
| | | **dev_id (LSB)** | **parm_type** | **opt** | **crc16 (H)** | **crc16 (L)** | | |
| 39 | 36 | 00 | 80 | 00 | OB | 8D | | |

**start flag**

start flagsof : 0x55AA

**message header**

len : 0x001E

cmd : 0x8009

seq : 0x00000003

pro_ver : 0x0001 (V0.1)

seq_flag : 0x0000

dev_id : 38 36 31 36 39 34 30 33 34 32 30 35 38 39 36 00 （in string: “861694034205896”）

**message content**

**parm_type** : 80 **Platform confirmation message**

opt : 00 Do nothing

**check**

crc16 : 0x0B8D

## 7.7.3 Tag Message TM Description

A message sent to the tag is a TM. The data format of the TM is described as follows

| SN | Field Name | Btye | Value description |
|---|---|---|---|
| 1 | Tag type | 1 | When 0xFF, it is a broadcast message, it will send a message to all tags |
| 2 | Tag ID | 4 | When it is 0xFF 0xFF 0xFF 0xFF, it is a broadcast message, and it will send a message to all tags of the same type |
| 3 | OPT | 1 | 0x00 No operation<br>0x01 Delete all local messages about Tag ID (excluding this message)<br>0x02 Delete all local messages about Tag Type (excluding this message) |
| 4 | msg sn | 1 | Value range (0x01 ~ 0xFF) |
| 5 | timeout | 1 | Unit: second (value range 10 to 255) |
| 6 | msg len | 1 | Unit: byte (value range 0 ~ 128) |
| 7 | msg | xx | The content of the message sent to the tag (the message length is determined by msg len) |

eg: 20 16 10 F4 B8 00 01 3C 0C 48 6F 77 20 61 72 65 20 79 6F 75 3F

| Tag type (1byte) | Tag ID (4bytes) | OPT (1byte) | msg sn (1byte) | timeout (1byte) | mag len (1Byte) | msg (12Bytes) |
|---|---|---|---|---|---|---|
| 20 | 16 10 F4 B8 | 00 | 00 | 3C | 0C | 48 6F 77 20 61 72 65 20 79 6F 75 3F |

Tag type : 20 student card

Tag ID : 16 10 F4 B8

OPT : 00 0x00 no operation

msg sn : 01 Message sequence number is 1

timeout : 3C That is, after this message is sent to the antenna, 60 seconds time out

msg len : 0C The length of the tag message content is 0x0C, which is 12Bytes

msg : 48 6F 77 20 61 72 65 20 79 6F 75 3F : tag message "How are you?"

# 8. TLV index

## 8.1 TLV type list and format

### 8.1.1 Format

| TLV type(2bytes) | LENGTH(2bytes) | VALUE(Length is defined by LENGTH） |
|---|---|---|

### 8.1.2 Type index

| Type | TAG | LENGTH | VALUE |
|---|---|---|---|
| RFID assets management | 0x8801 | 17 | Antenna Channel(1byte) + tag type(1byte) + id(4bytes) + sum(1byte) + activation address (2bytes) + tag status (1Bytes)+rssi(1byte)+reception time (6bytes) |
| current monitoring tag | 0x8901 | 16 | 1 byte signal strength + 4 bytes tag ID + 5 bytes tag sensing information + 6 bytes acquisition time |
| bracelet | 0x8A01 | 18 | 1 byte signal strength + 4 bytes bracelet ID + 1 byte type + 2 bytes data + 6 bytes collection time + 6 bytes reception time |
| Active tag (normal) | 0x8B01 | 17 | Antenna Channel(1byte)+tag type(1byte) + id(4bytes) + sum(1byte) + activation address/carrying information (2bytes)+voltage status (1Bytes)+rssi(1byte) +reception time (6bytes) |
| attendance tag | 0x8B02 | 17 | Time Attendance/Antenna Channel(1byte)+Tag Type(1byte)+id(4bytes)+sum(1byte)+Excitation Address/Carrying Information (2bytes)+Voltage Status (1Bytes)+rssi(1byte) +Reception Time (6bytes) |
| MR1054X tag | 0x8B03 | 23 | Antenna Channel(1byte)+Tag ID(15byte) +rssi(1byte) +Reception time(6bytes) |

normal tag eg1： 8B 01 00 11 01 20 78 2B 6A A4 2F 00 00 00 A9 11 01 0E 13 26 09

| TLV type (2bytes) | LENGTH(2bytes) | VALUE |
|---|---|---|
| 8B 01 | 00 11 | 01 20 78 2B 6A A4 2F 00 00 00 A9 11 01 0E 13 26 09 |

Analyze as follows：

TLV type : 0x8B01 normal tag

TLV data length : 0x0011

TLV date : 01 20 78 2B 6A A4 2F 00 00 00 A9 12 0B 0D 09 1C 31（For the format, please refer to the electronic tag data format description)

attendance tag eg2： 8B 02 00 11 81 20 78 2B 6A A4 2F 00 00 00 A9 11 01 0E 13 26 09

| TLV type (2bytes) | LENGTH(2bytes) | VALUE |
|---|---|---|
| 8B 02 | 00 11 | 81 20 78 2B 6A A4 2F 00 00 00 A9 11 01 0E 13 26 09 |

Analyze as follows：

TLVtype : 0x8B02 Attendance tag

TLV data length : 0x0011

TLV data : 81 20 78 2B 6A A4 2F 00 00 00 A9 11 01 0E 13 26 09 （For the format, please refer to the electronic tag data format description)

## 8.2TLV--Tag format description

Active tag（type is 0x8B01）Format description (total 17 bytes).

### 8.2.1 2.4G Active tag data format

This is mainly the data format of 2.4G active electronic tags (such as student cards, electric car tags, etc.).

Tag data forma：

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| **Antenna channel (1byte)** | **tag type (1byte)** | **tag ID (4bytes)** | **ID checksum (1byte)** | **Carry information (2bytes)** | **Tag status (1byte)** | **RSSI (1byte)** | **Receive time (6bytes)** | **description** |
| | **0x20** | | | activation address | | | | **student card** |
| | **0x21** | | | current info | | | | **current monitoring**<br><br>(The information carried is current, unit: mA, the first byte is the high byte.<br>eg 0x23 0x22 -> 0x2322 -> in DEC is 8994mA) |
| | **0x30** | | | activation address | | | | **safe city E-bike card** |
| | **0x31** | | | activati | | | | **safe city key chain tag** |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | on addres s | | | | |
| | **~~0x41~~** | | | ~~temper ature info~~ | | | | **~~temperature measuring tag~~**<br><br>~~（The information carried is the temperature, the first byte is the integer part of the temperature; the second byte is the fractional part of the temperature.~~<br>~~eg: 0x24 0x56 -> 36.86℃）~~ |
| | **0x42** | | | temper ature info | | | | **温 度 标 签 temperature measuring tag**<br>(The information carried is temperature, which is an unsigned number, with high byte first and low byte after; The actual temperature needs to be divided by 100 Unit: Degree（℃）Range: 0～44.99℃（ When the sensor is abnormal, it is 0xFFFF, after conversion, it is 655.35) eg: 0x0E 0x89 -> 0x0E89 (3721) -> 37.21℃) |
| | **0x50** | | | activati on addres s | | | | **assets management** |
| | | | | | | | | |

**Note：**

1. The ID checksum is the checksum of the tag type + tag ID (total 4 bytes) (see chapter 11.2 for the calculation method);

2. Antenna channel number description：

The highest bit (7bit): the state of entering and leaving the base station;

The lower four bits (0～3bit): the antenna number is 1,2,3,4 corresponding to the four antennas of east, south, west and north respectively)

| **In and out of the base station status/antenna channel** | | | |
|---|---|---|---|
| **7 bit** | **6 bit** | **5,4 bit** | **3～0 bit** |
| in/ou status： | Reader stay | reserved | antenna channel |

| | | | |
|---|---|---|---|
| 1 : enter reader RF area<br>0 : exit reader RF area | identification:<br>1 : Staying<br>0 : non-staying | | |

eg: 0x01 ， exit reader RF area, detected by antenna 1

***note:***

*1>. udging that entering and exiting the base station needs to consider both 7bit and 6bit;*

*2>. When the base station stay flag is 1, the status of entering and leaving the base station is invalid.*

3. Tag status description:

| Tag status byte format description | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **type** | **7bit** | **6bit** | **5bit** | **Tamper evident 4bit** | **3bit** | **2bit** | **1bit** | **Low voltage warning sign 0bit** |
| | | | | 1: remove alarm<br>0: normal | | | | 1: low voltage<br>0: normal |

E-bike tag/ student card eg1 : 01 20 78 2B 6A A4 2F 00 00 00 A9 12 0B 0D 09 1C 31

| antenna channel NO (1byte) | tag type (1byte) | Tag ID (4bytes) | ID checksum (1byte) | carried info (2bytes) | Tag status (1byte) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|---|---|---|
| 01 | 20 | 78 2B 6A A4 | 2F | 00 00 | 00 | A9 | 12 0B 0D 09 1C 31 |

explain:

01 : 0x01 ， exit reader RF area, detected by antenna 1

20 : **tag type, student card**;

78 2B 6A A4 : tag ID: 78 2B 6A A4 -> 0x782B6AA4 in DEC 2016111268

2F : 20 78 2B 6A A4checksum

00 00 : carried info, activation address 0x0000:Non-effective activation

00 : tag status, normal

A9 : Signal strength, -87dBm (single byte have signed number)

12 0B 0D 09 1C 31 : Tag receiving (reading) time, respectively year, month, day, hour, minute, second, the year is based on 2000, November 13, 2018 09:28:49

current monitoring tag eg2 : 01 21 78 2B 6A A4 2E 23 22 00 A9 12 0B 0D 09 1C 31

| antenna channel NO (1byte) | tag type (1byte) | tag ID (4bytes) | IDchecksum (1byte) | carried info (2bytes) | Tag status (1byte) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|---|---|---|
| 01 | 21 | 78 2B 6A A4 | 2E | 23 22 | 00 | A9 | 12 0B 0D 09 1C 31 |

explain:

01 : 0x01 ， exit reader RF area, detected by antenna 1

21 : current monitoring tag

78 2B 6A A4 : tag ID: 78 2B 6A A4 -> 0x782B6AA4 in DEC 2016111268

2E : 21 78 2B 6A A4 checksum

23 22 : carried info, current 0x2322, that is 8994mA

00 : tag status, normal

A9 : Signal strength, -87dBm (single byte have signed number)

12 0B 0D 09 1C 31 : Tag receiving (reading) time, respectively year, month, day, hour, minute, second, the year is based on 2000, November 13, 2018 09:28:49

temperature measuring tag eg3 : 84 42 78 2B 6A A4 0D 0E 24 80 A9 12 0B 0D 09 1C 31

| antenna channel NO (1byte) | tag type (1byte) | tag ID (4bytes) | IDchecksum (1byte) | carried info (2bytes) | Tag status (1byte) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|---|---|---|
| 84 | 42 | 78 2B 6A A4 | 0D | 0E 24 | 80 | A9 | 12 0B 0D 09 1C 31 |

explain:

84 : 0x84 , enter
(this byte: enter is 0x84, read from antenna4)

42 : temperature measuring tag

78 2B 6A A4 : tag ID: 78 2B 6A A4 -> 0x782B6AA4 in DEC 2016111268

0d : 42 78 2B 6A A4checksum

0E 24 : Temperature information, That is, 0x0E24 is converted to a decimal number as 3620, and converted to a temperature of 36.2℃

80 : tag status, voltage is normal, enter

A9 : Signal strength, -87dBm (single byte have signed number)

12 0B 0D 09 1C 31 : Tag receiving (reading) time, respectively year, month, day, hour, minute, second, the year is based on 2000, November 13, 2018 09:28:49

| Temperature measuring tag status byte format description | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| type | 7bit | 6bit | 5bit | Tamper evident 4bit | 3bit | 2bit | 1bit | Low voltage warning sign 0bit |
| | 1: enter<br>0: exit | | | 1: remove alarm<br>0: normal | | | | 1: Low voltage alarm<br>0: normal |

## 8.2.2 WiFi probe data format

This refers to uploading the read WiFi MAC address data, and the tag type is fixed at 0x90

eg : 82 90 84 3A 4B 61 DB 48 00 00 B1 13 05 05 0B 0E 2E

| Antenna channel NO (1byte) | Tag type (1byte) | WiFi Mac address (6bytes) | Reserve (2bytes) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|---|
| 82 | 90 | 84 3A 4B 61 DB 48 | 00 00 | B1 | 13 05 05 |

| | | | | | |
|---|---|---|---|---|---|
| | | | | | 0B 0E 2E |

explain：

82 ： **enter, exit reader status/ antenna**（Channel） this byte:

The highest bit (7bit): the state of entering and leaving the base station;

The lower four digits (0～3bit): the antenna number is 1,2,3,4 corresponding to the four antennas of east, south, west and north respectively)

| **In and out of the base station status/antenna channel** | | | |
|---|---|---|---|
| **7 bit** | **6 bit** | **5,4 bit** | **3～0 bit** |
| in/ou status：<br>1： enter reader RF area<br>0： exit reader RF area | Reader stay identification：<br>1 : Staying<br>0 : non-staying | reserved | antenna channel |

0x82 ， enter reader RF area, last detected by antenna 2

***note：***

*1>. udging that entering and exiting the base station needs to consider both 7bit and 6bit;*

*2>. When the base station stay flag is 1, the status of entering and leaving the base station is invalid.*

90 ： Tag type, WiFi probe data；

84 3A 4B 61 DB 48 ： WiFi Macaddress84-3A-4B-61-DB-48

00 00 ： reserved bytes

B1 ： Signal strength, -79dBm (single byte signed number)

13 05 05 0B 0E 2E ：Tag receiving (reading) time, respectively year, month, day, hour, minute, second, year is based on 2000, May 05, 2019 11:14:46

## 8.3TLV--Attendance tag format description

Attendance tag (type 0x8B02) format description (a total of 17 bytes).

### 8.3.1 2.4G Active electronic card tag data format

This is mainly the data format of 2.4G active electronic tags (such as self-operated student cards, temperature measurement tags, etc.).

Tag data format：

| 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
|---|---|---|---|---|---|---|---|---|
| **Antenna channel NO (1byte)** | **tag type (1byte)** | **tag ID (4bytes)** | **ID checksum (1byte)** | **Carried info (2bytes)** | **Tag status (1byte)** | **RSSI (1byte)** | **Time (6bytes)** | **description** |
| | **0x40** | | | | | | | self-operated student card |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | reserved | | | | |
| | ~~**0x41**~~ | | | ~~temperature info~~ | | | | ~~temperature measurement tag~~<br><br>~~（The information carried is the temperature, the first byte is the integer part of the temperature; the second byte is the fractional part of the temperature。~~<br>~~eg: 0x24 0x3C -> 36.5℃）~~ |
| | **0x42** | | | temperature | | | | (The information carried is temperature, which is an unsigned number, with high byte first and low byte after; The actual temperature needs to be divided by 100 Unit: Degree (℃) Range: 0～44.99℃（ When the sensor is abnormal, it is 0xFFFF, after conversion, it is 655.35) eg: 0x0E 0x89 -> 0x0E89 (3721) -> 37.21℃) |

**Note：**

1. The ID checksum is the checksum of the tag type + tag ID (see chapter 11.2 for the calculation method);

2. Antenna channel number description：

The highest bit (7bit): the state of entering and leaving the base station;

The lower four bits (0～3bit): the antenna number is 1,2,3,4 corresponding to the four antennas of east, south, west and north respectively)

| **In and out of the base station status/antenna channel** | | | |
|---|---|---|---|
| **7 bit** | **6 bit** | **5,4 bit** | **3～0 bit** |
| in/ou status：<br>1： enter reader RF area<br>0： exit reader RF area | Unilateral attendance identification:<br>1: Unilateral Time Attendance<br>0: Not unilateral attendance | reserved | antenna channel |

eg: 0x**01** ， exit reader RF area, last detected by antenna 2

***note：***

*1>. udging that entering and exiting the base station needs to consider both 7bit and*

*6bit;*

*2>. When the base station stay flag is 1, the status of entering and leaving the base station is invalid.*

3. Tag status description：

| Tag status byte format description | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **type** | **7bit** | **6bit** | **5bit** | **Tamper evident 4bit** | **3bit** | **2bit** | **1bit** | **Low voltage warning sign 0bit** |
| | | | | 1: remove alarm<br>0: normal | | | | 1: low voltage<br>0: normal |

self-operated student card eg 1: 81 40 **78 2B 6A A4** **0F** **00 00** 00 A9 11 01 0E 13 26 09

| Attendance /antenna channel NO (1byte) | tag type (1byte) | tag ID (4bytes) | ID checksum (1byte) | carried info (2bytes) | Tag status (1byte) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|---|---|---|
| 81 | 40 | **78 2B 6A A4** | **0F** | 00 00 | 00 | **A9** | 11 01 0E 13 26 09 |

81 : enter reader RF area, detected by antenna 1

**40** : tag type, self-operated student card

**78 2B 6A A4** : tag ID

**0F** : 40 **78 2B 6A A4 checksum**

**00 00** : reserved

00 : No voltage alarm (low voltage is 0x01, normal is 00)

**A9** : Signal strength, -87dBm (single byte signed number)

11 01 0E 13 26 09 : attendance time, respectively year, month, day, hour, minute, second, year is based on 2000, January 14, 2017 19 : 38 : 09

Temperature measurement tag eg2 : **82 42 78 2B 6A A4 0D 0E 24** 80 **A9** 12 0B 0D 09 1C 31

| antenna channel NO (1byte) | tag type (1byte) | tag ID (4bytes) | IDchecksum (1byte) | carried info (2bytes) | Tag status (1byte) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|---|---|---|
| **82** | **42** | **78 2B 6A A4** | **0D** | **0E 24** | 80 | **A9** | 12 0B 0D 09 1C 31 |

explain：

**82** : 0x**82** ， enter school It was finally read by Antenna 2.

**42** : Temperature measurement tag

**78 2B 6A A4** : Tag ID: **78 2B 6A A4** -> 0x782B6AA4 in DEC 2016111268

**0D** : **42 78 2B 6A A4** checksum

**0E 24** : Temperature information, That is, 0x0E24 is converted to a decimal number as

3620, and converted to a temperature of 36.2℃

80 : tag status, normal voltage, enter the door

A9 : Signal strength, -87dBm (single byte signed number)

12 0B 0D 09 1C 31 : Tag receiving (reading) time， respectively year, month, day, hour, minute, second, year is based on 2000, November 13, 2018 09 :28 : 49

| Temperature measuring tag status byte format description | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **type** | **7bit** | **6bit** | **5bit** | **Tamper evident 4bit** | **3bit** | **2bit** | **1bit** | **Low voltage warning sign 0bit** |
| | 1：enter<br>0：exit | | | 1: remove alarm<br>0: normal | | | | 1: Low voltage alarm<br>0: normal |

## 8.4TLV--Assets management tag format description

Assets management tag (type 0x8801) format description (a total of 17 bytes).

eg : 01 50 E3 AF 22 32 CA 00 00 00 B2 11 01 0E 13 26 09

| antenna channel NO (1byte) | tag type (1byte) | tag ID (4bytes) | ID checksum (1byte) | activation address (2bytes) | Tag status (1byte) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|---|---|---|
| 01 | 50 | E3 AF 22 32 | FA | 00 00 | 00 | B2 | 11 01 0E 13 26 09 |

explain：

01 : **Antenna number, generally an antenna in the asset management system, fill in 1.**

50 : The tag type is an asset tag. It is not recommended that the software use this mark to distinguish the tag type. It should be distinguished by the upper level TAG (TLV type 0x8801);

E3 AF 22 32 : Tag ID

CA : 50 E3 AF 22 32 checksum

00 00 : activation address, non-activation

00 : tag status, normal

B2 : Signal strength, -87dBm (single byte signed number)

11 01 0E 13 26 09 : Tag receiving (reading) time， respectively year, month, day, hour, minute, second, year is based on 2000, January 14, 2017, 19 : 38 : 09

Tag status description：

| Tag status byte format description | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **type** | **7bit** | **6bit** | **5bit** | **Tamper evident 4bit** | **3bit** | **2bit** | **1bit** | **Low voltage warning sign 0bit** |

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | 1: remove alarm<br>0: normal | | | | 1: low voltage<br>0: normal |

## 8.5TLV--MR1054X tag format description

Electronic tag (type 0x8B03) format description (24 bytes in total).

eg : 04 **01** **39 39 39 32 36 31 30 30 30 30 30 30 30 30 34** **E1** 13 04 19 0E 30 2A

| antenna channel no (1byte) | tag type (1byte) | tag ID (15bytes) | RSSI (1byte) | Time (6bytes) |
|---|---|---|---|---|
| 04 | 01 | **39 39 39 32 36 31 30 30 30 30 30 30 30 30 34** | **E1** | 13 04 19 0E 30 2A |

explain：

**04** ： **antenna**（Channel） this byte:

The highest bit (7bit): the state of entering and leaving the base station;

The lower four digits (0～3bit): the antenna number is 1,2,3,4 corresponding to the four antennas of east, south, west and north respectively)

| In and out of the base station status/antenna channel | | | |
|---|---|---|---|
| **7 bit** | **6 bit** | **5,4 bit** | **3～0 bit** |
| in/ou status：<br>1： enter reader RF area<br>0： exit reader RF area | Reader stay identification：<br>1 : Staying<br>0 : non-staying | reserved | antenna channel |

0x**04** ， exit reader RF area, last detected by antenna 4

***note：***

*1>. udging that entering and exiting the base station needs to consider both 7bit and 6bit;*

*2>. When the base station stay flag is 1, the status of entering and leaving the base station is invalid.*

**01** ： Tag type (reserved bytes, no actual definition)

**39 39 39 32 36 31 30 30 30 30 30 30 30 30 34**

： 15 byte tag ID

**E1** ： Signal strength, -87dBm (single byte signed number)

13 04 19 0E 30 2A ： tag reading time respectively year, month, day, hour, minute, second, year is based on 2000, April 25, 2019, 14 : 48 : 42

# 9.Configuration parameter format description

The following is a description of the configuration parameter format of the device. User parameters do not include items 30 to 34 of the serial number.

**explain：**

R : Read ONLY

R/W : Reader & Write

| SN | Data file | Bytes | R/W | Type | Description |
|---|---|---|---|---|---|
| 1 | Parameter eigenvalue | 1 | R/W | value | When writing, keep 0x55 unchanged, the parameters can be configured normally,<br>When writing, if it is not 0x55, the factory default parameters will be restored;<br>When reading, fixed at 0x55 |
| 2 | Communication mode | 1 | R/W | value | Low 4 digits, read only, transmission method:<br>Bit 1: read only, GPRS connection, 1 valid<br>2nd bit: read only, LAN connection, 1 valid<br>Bit 3: reserved<br>4th bit: reserved<br>5th bit: readable and writable, tag transmission identification<br>0: transmit the tag record to the platform;<br>1: Does not transmit tag records to the platform<br>6th bit: readable and writable, encrypted logo<br>0: No encryption, that is, the message content is plain text;<br>1: Encryption, that is, message content is cipher text<br>7th～8bit: reserved<br>eg: 0x11, which means GPRS connection, no tag record is transmitted to the platform.<br>eg: 0x01, means GPRS connection, transfer the tag record to the platform |
| 3 | Firmware version | 2 | R | value | Firmware version, major version number first<br>eg: 01 05， V1.5 |
| 4 | buzzer | 1 | R/W | value | 1：ON，0：OFF |
| 5 | GPRS module type | 1 | R | value | 0x00 : SIM800C,<br>0x01: SIM7600CE<br>0x02: EC20 |
| 6 | Deduplication window (Judgment time to leave | 2 | R/W | value | Tag deduplication filter window, unit: second; no filter at 0x0000, low byte first, high byte second<br>Value range: 20～65535 |

| | | | | | |
|---|---|---|---|---|---|
| | the base station) | | | | |
| 7 | Device ID | 16 | R/W | string | The 15-digit product serial number is ASCII, and the last 1 byte is filled with 0x00<br>eg: “861694034205896” |
| 8 | The base station stays over time | 2 | R/W | value | The tag stays at the base station and is reported to the platform according to the interval of this value. The low byte is first and the high byte is next.<br>Unit: second<br>Value range: 0, 60～65535<br>0: Do not use this function<br>60～65535: timeout time |
| 9 | DHCP enable | 1 | R/W | value | 1：Open， 0：Closed，for LAN |
| 10 | LAN local IP | 4 | R/W | value | Used for LAN network parameter configuration |
| 11 | LAN Mask | 4 | R/W | value | Used for LAN network parameter configuration |
| 12 | LAN gateway | 4 | R/W | value | Used for LAN network parameter configuration |
| 13 | LAN local port | 2 | R/W | value | Local IP port, applicable to LAN, low byte first high byte after (value range 0 ~ 65536)<br>eg: *24 13 , is 0x1324(HEX) = 4900 (DEC)* |
| 14 | GPRS server 1 IP | 32 | R/W | string | IP or domain name, string |
| 15 | G GPRS server 1 port | 2 | R/W | value | low byte before, high byte after (value range 0 ~ 65536) |
| 16 | LAN server 1 IP | 32 | R/W | string | IP or domain name, string |
| 17 | LAN server 1 port | 2 | R/W | value | low byte before, high byte after (value range 0 ~ 65536) |
| 18 | LAN local mac | 6 | R/W | value | Used for LAN network parameter configuration |
| 19 | Reserved 1 | 28 | - | - | - |
| 20 | Antenna version | 8 | R | value | Corresponding to the firmware version of 4 antennas<br>Each antenna version occupies 2 bytes<br>eg: 02 00 is V2.0, if it is FF FF means reading the antenna version failed (may be no antenna connected) |
| 21 | GPRSSignal strength | 1 | R | value | 00 or 99 means GPRS has no signal<br>99 means failed to read gprs signal<br>Normal value range 0～31 |
| 22 | Device ID | 16 | R | string | eg: “MR7901P-03C0025” |

| | | | | | |
|---|---|---|---|---|---|
| 23 | Antenna RSSI threshold | 4 | R/W | Signed value | Value range: -1 ~ -128<br>The signal strength RSSI filter threshold of 1,2,3 and 4 corresponding to the antenna is respectively.<br>eg: -88 |
| 24 | Antenna gain | 4 | R/W | value | The value range is 0~ 31, corresponding to the signal gain value of the antenna 1,2,3 and 4 |
| 25 | Bluetooth output tag identification | 1 | R/W | value | 0: Cannot output tags via Bluetooth<br>1: The tag can be output through Bluetooth. At this time, the tag read by the antenna will only be output through Bluetooth, and the output of the tag is prohibited by gprs and lan. |
| 26 | Communicate status | 1 | R | value | Direct connection status with the platform<br>Lower 4 digits, connection method<br>Bit 1: read only, GPRS connection, 1 valid<br>2nd bit: read only, LAN connection, 1 valid<br>Bit 3: reserved<br>4th bit: reserved<br>High 4 digits, fixed at A<br>eg:<br>0xA0 no connection<br>0xA1 has GPRS connection<br>0xA2 with LAN connection<br>0xA3 with GPRS, LAN connection |
| 27 | Tag filter identification | 1 | R/W | value | 1: Use filtering (ie, direction judgment filtering)<br>0: No filtering is used (upload in, out) |
| 28 | tag filtering interval | 1 | R/W | value | Unit: Second |
| 29 | reserved 2 | 2 | - | - | |
| User parameters do not include the following | | | | | |
| 30 | IP of server 2 | 32 | R/W | string | IP or domain name, string |
| 31 | Port of server 2 | 2 | R/W | value | Low byte first, high byte last |
| 32 | Antenna firmware upgrade sign | 1 | R | value | 0xA5 upgrade file is valid<br>Other values, no valid upgrade file |
| 33 | Antenna firmware upgrade file size | 4 | R | value | Low byte first<br>If the device does not store the upgrade file, it is 0x000000000<br>eg: 0x0000288E,The upgrade file size is 10382Bytes |
| 34 | Antenna firmware | 2 | R | value | if the device does not store the upgrade file, value is 0000 |

| | | | | | |
|---|---|---|---|---|---|
| | upgrade file version | | | | eg ： 02 00， version is V2.0 |
| 35 | Antenna firmware upgrade file crc16 check | 2 | R | value | if the device does not store the upgrade file, value is 0000<br>Upgrade file CRC16 check value (temporarily useless) |
| 36 | Reserved 3 | 15 | - | - | |

# 10. Command summary

| SN | cmd | Description |
|---|---|---|
| 1 | 0x0008 | Terminal (Device/Reader) registration request |
| 2 | 0x8008 | Platform confirmation code for terminal registration |
| 3 | 0x0001 | Terminal login request |
| 4 | 0x8001 | Platform confirmation code for terminal login |
| 5 | 0x0003 | Terminal sends heartbeat |
| 6 | 0x8003 | Platform confirmation code forTerminal sends heartbeat |
| 7 | 0x0004 | Terminal sends tag data |
| 8 | 0x8004 | Platform confirmation code forTerminal sends tag data |
| 9 | 0x000D | Upgrade firmware |
| 10 | 0x800D | Platform confirmation code for Upgrade firmware |
| 11 | 0x000A | Report configuration parameters |
| 12 | 0x800A | Platform confirmation code for Report configuration parameters |
| 13 | 0x0009 | Request tag message |
| 14 | 0x8009 | The platform responds to the tag message |

# 11. Check algorithm

## 11.1 CRC16Check algorithm

The data packets communicated with the platform are checked by CRC16. Two verification algorithms are introduced below. Method 2 is recommended. Method 2 uses a table lookup method, which is 8 times faster than Method 1.

### 11.1.1 C language function method 1

```
/*****************************************************************************
** Function name       :    crc16_ccitt
** Descriptions        : Cyclic Redundancy Check-16 (CCITT Standard-0x1021)
** input parameters    :    buf    Data to be verified
**                              len  Length of check data
** output parameters :    none
** Returned value      :     Check value
*****************************************************************************/
uint16_t crc16_ccitt(uint8_t *buf, uint16_t len)
{
    uint16_t i, j;
    uint16_t crc_reg = 0xFFFF;
    uint16_t crc_val;

    for (i = 0; i < len; i++)
    {
        crc_val = buf[i] << 8;

        for (j = 0; j < 8; j++)
        {
            if (((int16_t)(crc_reg ^ crc_val)) < 0)
                crc_reg = (crc_reg << 1) ^ 0x1021;
            else
                crc_reg <<= 1;
            crc_val <<= 1;
        }
    }
    return crc_reg;
```

```
}
```

## 11.1.2 C language function method 2: Look-up table method

```
/*****************************************************************************
** Function name      :  CRC16
** Descriptions       :   Cyclic Redundancy Check-16 (CCITT Standard-0x1021)
** input parameters   :  Data Data to be verified
**                          Length Length of check data
** output parameters :  none
** Returned value     :   check value
*****************************************************************************/
const uint16_t crc16_table[]=     //* CRC16 CCITT standard -0x1021
{
    0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7,
    0x8108, 0x9129, 0xa14a, 0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef,
    0x1231, 0x0210, 0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
    0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c, 0xf3ff, 0xe3de,
    0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485,
    0xa56a, 0xb54b, 0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
    0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6, 0x5695, 0x46b4,
    0xb75b, 0xa77a, 0x9719, 0x8738, 0xf7df, 0xe7fe, 0xd79d, 0xc7bc,
    0x48c4, 0x58e5, 0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
    0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969, 0xa90a, 0xb92b,
    0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a12,
    0xdbfd, 0xcbdc, 0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
    0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03, 0x0c60, 0x1c41,
    0xedae, 0xfd8f, 0xcdec, 0xddcd, 0xad2a, 0xbd0b, 0x8d68, 0x9d49,
    0x7e97, 0x6eb6, 0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
    0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a, 0x9f59, 0x8f78,
    0x9188, 0x81a9, 0xb1ca, 0xa1eb, 0xd10c, 0xc12d, 0xf14e, 0xe16f,
    0x1080, 0x00a1, 0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
    0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c, 0xe37f, 0xf35e,
    0x02b1, 0x1290, 0x22f3, 0x32d2, 0x4235, 0x5214, 0x6277, 0x7256,
    0xb5ea, 0xa5cb, 0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
    0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,
    0xa7db, 0xb7fa, 0x8799, 0x97b8, 0xe75f, 0xf77e, 0xc71d, 0xd73c,
    0x26d3, 0x36f2, 0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
    0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9, 0xb98a, 0xa9ab,
    0x5844, 0x4865, 0x7806, 0x6827, 0x18c0, 0x08e1, 0x3882, 0x28a3,
    0xcb7d, 0xdb5c, 0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
```

```
    0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0, 0x2ab3, 0x3a92,
    0xfd2e, 0xed0f, 0xdd6c, 0xcd4d, 0xbdaa, 0xad8b, 0x9de8, 0x8dc9,
    0x7c26, 0x6c07, 0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
    0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba, 0x8fd9, 0x9ff8,
    0x6e17, 0x7e36, 0x4e55, 0x5e74, 0x2e93, 0x3eb2, 0x0ed1, 0x1ef0
};

uint16_t   CRC16(uint8_t   * Data,   uint16_t   Length)
{
    uint16_t   crc;
    uint8_t   da;

        crc = 0xFFFF;
        while(Length--!=0)
        {
            da=(uint8_t) (crc/256);
            crc <<= 8;
            crc ^= crc16_table[da^*Data];
            Data++;
        }

    return crc;
}
```

## 11.1.3 JAVA

```
/**************************************************************************
 *CRC-CCITT standard calculation   JAVA
 *
 *
 *input：Byte array data need to be check
 *
 *output：Return 2 bytes of hexadecimal check code
 *************************************************************************/
private static String getCrc(byte[] data) {
    int crc = 0xFFFF;              //crc Calculate the initial value

    for (int i = 0; i < data.length; i++) {
        crc = (data[i] << 8) ^ crc;
        for (int j = 0; j < 8; ++j) {
          if ((crc & 0x8000) != 0)
                crc = (crc << 1) ^ 0x1021;
          else
```

```
                    crc <<= 1;
                }
            }
            return Integer.toHexString(crc & 0xFFFF).toUpperCase();
        }
```

## 11.2 Checksum algorithm

Checksum is used in tag data.

### 11.2.1 C language

```
/*****************************************************************************
** Function name       :   CheckSum
** Descriptions        :   checksum
** input parameters    :   uBuff Data to be verified
**                         uBuffLen    ength of check data
** output parameters :   none
** Returned value     :    check value
*****************************************************************************/
  uint8    CheckSum(uint8 *uBuff, uint16 uBuffLen)
  {
      uint16 i;
      uint8 uSum=0;

      for(i=0;i<uBuffLen;i++)
      {
          uSum = uSum + uBuff[i];
      }
      uSum = (~uSum) + 1;

      return uSum;
  }
```

### 11.2.2 JAVA

```
/*********************************************************
 * Checksum JAVA algorithm in tag data
 *
 * @param sendbyte    The checksum interval needs to be calculated: 1 byte tag TYPE + 4 byte tag ID
 *
```

```
 * @return            Calculated checksum
 *                      20 E3 AF 22 32 checksum：FA(in DEC is  -6）
**********************************************************/
protected static byte sendRcvByteNum(byte[] sendbyte) {
        byte sum = 0;
        for (int i = 0; i < sendbyte.length; i++) {
            sum += sendbyte[i];
        }
        byte rebyte = (byte) (~sum + 1);
        System.out.println("Check Digit：" + rebyte);
        return rebyte;
    }

//Calculated checksum 20 E3 AF 22 32 checksum ： FA(in DEC is  -6）
public static void main(String[] args) {
        byte[] b = new byte[5];
        b[0] = 0x20;
        b[1] = (byte) 0xE3;
        b[2] = (byte) 0xAF;
        b[3] = 0x22;
        b[4] = 0x32;
        sendRcvByteNum(b);
    }
```

# 12. Appendix

## 12.1 Correspondence between battery power and battery voltage

The correspondence between the battery voltage on the MR7901 and the battery power in the heartbeat message (or uploading the device hardware information) is as follows:

| battery power | Actual battery voltage (unit V) |
|---|---|
| 10 | ≥ 8.00 |
| 9 | ≥ 7.75, ＜8.00 |
| 8 | ≥ 7.50, ＜7.75 |
| 7 | ≥ 7.25, ＜7.50 |
| 6 | ≥ 7.00, ＜7.25 |
| 5 | ≥ 6.75, ＜7.00 |
| 4 | ≥ 6.50, ＜7.75 |
| 3 | ≥ 6.25, ＜6.50 |
| 2 | ≥ 6.00, ＜6.25 |
| 1 | ≥ 5.75, ＜6.00 |
| 0 | ＜5.75 |

explain：

1. The battery voltage is lower than 5.75V, that is, when the battery power is lower than 1, the device cannot work normally;

2. When the battery power is lower than 9, it can be judged that the external power supply has been cut off (or lost).

# 13. Security mechanism

In order to protect the platform from being accessed by illegal devices or links and cause damage to the platform security and stored data, it is necessary to adopt corresponding security mechanisms in the communication between the device and the platform.

## 13.1 Data interaction security between device and platform

Related instructions：

1. The data interaction security between the terminal and the platform is mainly achieved by encrypting the message content;
2. The encryption and decryption of the message content between the terminal and the platform is achieved using a symmetric AES algorithm, CFB mode, and a 128-bit secret key；
3. Both the terminal and the platform have 16 sets of secret keys, and decide which set of secret keys to use for encryption and decryption according to the value of the modulo 16 of the message serial number in the message header；
4. After the device leaves the factory or restores the factory parameters, the communication between the device and the platform uses plain text communication by default, and the local key of the device will also be restored to the 16 sets of factory secret keys；
5. Whether the message content of the communication between the device and the platform is encrypted is controlled by the "encrypted identification" bit in the device system parameters；
6. The encrypted initial vector IV is fixed, but the encryption key can be changed (modified by configuration parameter instructions)。

### 13.1.1 Encryption on or off

The factory default is to use plain text transmission. If business needs, you can enable the encryption function by configuring the "encryption flag" in the system parameters to achieve message content cipher text communication。

There are two ways to modify (enable or disable) the encrypted logo. The operation flow is described as follows：

**1. Open by configuring system parameters**

1>. The platform starts the process of configuring system parameters when responding to device login and heartbeat；

2>. For the position of the encryption mark in the system parameters, please refer to the 6th bit of the second byte of the "Communication Mode" in Chapter 9 "Configuration Parameter Format Description", which is the encryption mark。

**2. Open via the opcode carried in the heartbeat response**

The platform uses 0x0A (requires to update the encrypted transmission ID) in the heartbeat opcode of the reply device, that is, to enter the process of separately configuring the "encrypted ID"。

### 13.1.2 Encryption

In encrypted communication, the AES algorithm CFB encryption mode is used, and the 128-bit key encrypts and decrypts the message content.

There are 16 sets of secret keys stored locally on the device. After encryption is enabled on the device, the message serial number is used to inform the other party which set of keys are currently being encrypted; the specific method: modulo 16 by the serial number to determine which set of keys。

### 13.1.3 Key modification

The process for the platform to modify the secret key: the secret key modification process is similar to the system parameter modification process, the specific process is as follows：

1. The platform informs the device through a heartbeat response that the platform needs to modify the device key；

2. After receiving the instruction, the device first uploads the local key；

3. After receiving the key uploaded by the device, the platform sends the new key to the device；

4. After receiving the newly issued key, the device first responds to the device and then uses the new key to start communication。

### 13.1.4 Starting key

1. The initial vector IV of the encrypted CFB is fixed to (hexadecimal)：

32 77 23 64 25 65 55 6D 38 5F 57 45 40 30 56 68

2. The factory default initial 16 sets of keys (hexadecimal) are：

| No. | Secret key |
|---|---|
| 0 | A1 B0 B4 F3 BC D2 BA C3 A3 A1 BD F1 CC EC A3 AC |
| 1 | CE D2 C3 C7 D5 D9 BF AA D5 E2 B8 F6 D7 F9 CC B8 |
| 2 | BB E1 A3 AC D6 F7 D2 AA CA C7 CC FD CC FD B4 F3 |
| 3 | BC D2 B6 D4 BE AD BC C3 B7 A2 D5 B9 D0 CE CA C6 |
| 4 | BA CD C3 F1 D3 AA BE AD BC C3 B7 A2 D5 B9 B5 C4 |
| 5 | D2 E2 BC FB BA CD BD A8 D2 E9 A1 A3 CA D7 CF C8 |
| 6 | A3 AC CE D2 CF F2 D4 DA D7 F9 B8 F7 CE BB C3 F1 |
| 7 | D3 AA C6 F3 D2 B5 BC D2 BA CD C8 AB B9 FA B9 E3 |
| 8 | B4 F3 C3 F1 D3 AA C6 F3 D2 B5 BC D2 A3 AC D6 C2 |
| 9 | D2 D4 B3 CF D6 BF B5 C4 CE CA BA F2 A3 A1 A1 B1 |
| 10 | 0D 0A 0D 0A A1 A1 A1 A1 CF B0 BD FC C6 BD D7 DC |
| 11 | CA E9 BC C7 B5 C4 BF AA B3 A1 B0 D7 BF AA C3 C5 |
| 12 | BC FB C9 BD A1 A2 D6 B1 B1 BC D6 F7 CC E2 A3 AC |
| 13 | D3 AE B5 C3 B2 CE BB E1 C3 F1 D3 AA C6 F3 D2 B5 |
| 14 | BC D2 B4 FA B1 ED B5 C4 C8 C8 C1 D2 D5 C6 C9 F9 |
| 15 | A1 A3 0D 0A 0D 0A A1 A1 A1 A1 C3 F1 D3 AA BE AD |