**Oviphone Technology Limited Company: Z66-4G-CAT1 device TCP protocol**

**The server needs to respond with F0 login packet (with specific response format) and F9 heartbeat packet (no specific response format), otherwise itl.**

Catalogue

1. Overview 1

2. Protocol Data Packet Structure 1

2.1. Data Header 1

2.2. Message ID 2

2.3. Token Generation Mechanism 2

2.4. Payload 2

2.5. Checksum 3

3.messages 4

3.1 Connection related 4

3.1.1 LNK-LIN (0xF0) Request Connection (TCP Only) 4

3.1.2LNK-RPL(0xF1) Connection Reply (TCP only) - Important 5

3.1.3New heartbeat packet protocol (0xF9) - Important 6

3.1.4Heartbeat protocol (0xF6) (used in previous devices, will not coexist with F9) 7

3.2 Positioning related reporting 8

3.2.1GPS/BDS Position Reporting: Location Data Reporting (0x03) 8

3.2.2Upload of Wi-Fi and base station information (0xA4 improved version) 10

3.2.3 Bluetooth positioning information(LBE Location)（MsgId=0xD6） 12

3.3 Alarm related reporting 14

3.3.1Alarm data upload (0x02) 14

3.3.2Alarm data upload(0x21) (supplement to 0x02) 16

3.4Equipment information and status reporting 17

3.4.1Status parameter reporting (MSGID=0xA9) - no need for parsing 17

3.4.2 Upload of software version and model（0XBB）- no need for parsing 18

3.1.2 3.4.3 ICCID upload of SIM card (0xF3) 19

3.5 Downstream feedback report 19

3.5.1Downlink feedback(MSGID=0xC0) 19

3.5.2Message status reporting(MSGID=0x28) 20

4.Setting 21

4.1Downlink 21

4.1.1 Set periodic upload（0x17） 21

4.1.2Message Send（MSGID=0X28） 22

4.1.3Setting（0XCE）---See detailed instructions-Important Downside 23

#

# Overview

This agreement is applicable to Oviphone Communication's Z66 4G CAT1 equipment, which currently supports products such as Z66 .

* Use 32-bit data headers for synchronization and terminal identification;
* Implement verification protection using low-cost verification algorithms;
* Use message identifiers to indicate different messages.
* Unless otherwise specified. All applicable.

# Protocol Data Packet Structure

A basic protocol data packet structure is shown in Figure 1.：



**Figure1**

## Data Header

 Each data packet begins with a 4-byte Header or token (in some response messages, a timestamp is used instead):

 Currently, the token for Eurofins devices is fixed as BD BDBDBD

Header: 0xBD 0xBD 0xBD 0xBD

Timestamp: 32 bits, generated by the server

## Message ID

 MessageId represents the content as described in Chapter 3.

 Every time the TCP connection is established, the device side will first report the 0xF0 message, which contains the unique identifier IMEI of the device. The server side needs to record this IMEI as an identifier. Then it will respond with the 0xF1 message. The device side will consider the connection successful only after receiving this response. Otherwise, it will disconnect the link.

## Token Generation Mechanism

Currently fixed to BDBDBDBD

## Payload

The payload below refers to the effective content of the protocol, excluding the head token and checksum. The length of the content is indicated afterwards.

The data formats used in the payload are shown in the following table:

 [U-unsigned; I-signed; X-bitfield; number-bytes occupied]

 In the protocol below, little-endian is used for all data types except for ch, u8, i8, and x8.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Short** | **peTypeType** | **Size(Bytes)** | **Min/max** | **Resolution** | **explain** |
| CH | ASCII/ISO 8859.1 | 1 | - | - | char |
| u8 | Unsigned Char | 1 | 0..255 | 1 | unsigned short |
| i8 | Signed Char | 1 | -128..127 | 1 | short |
| x8 | Bitfield | 1 | - | - | bit |
| u16 | Unsigned Short | 2 | 0..65，535 | 1 | unsigned int |
| i16 | Unsigned Short | 2 | -32,768..32,767 | 1 | int |
| x16 | Bitfield | 2 | - | - | （bit）2 |
| u32 | Unsigned Long | 4 | 0..4,294,967,295 | 1 | unsigned long |
| i32 | Signed Long | 4 | -2,147,483,648..2,147,483,647 | 1 | long |
| u64 | Uint64\_t | 8 | 0..18,446,744,073,709,551,616 | 1 | uint64\_t |
| float | float | 4 | -3.44\*10e38..3.4\*10e38 | - | float |

## Checksum

The content to be added to the checksum includes the payload, as shown in Figure 1. The algorithm is as follows, where Buffer[N] represents the data to be accumulated.

Ck\_sum = 0

For(i=0; i<N; i++)

{

ck\_sum = ck\_sum + Buffer[i]

ck\_sum = ck\_sum % 0x100

}

ck\_sum = 0xFF – ck\_sum

Return ck\_sum

Where ck\_sum cannot exceed 0xFF, so after each loop, it must be modulo 0x100 and then taken the remainder.

# **3.**messages

## Connection related

### LNK-LIN (0xF0) Request Connection (TCP Only)

|  |  |
| --- | --- |
| Message | LNK-LIN |
| Description | Terminal requests to log into the terminal server through IMEI number |
| Firmware | 　 |
| Direction | Terminal => Terminal Server |
| Payload length | 10bytes |
| Message structure | Header | Message ID | Payload | Checksum |
| 0xBD 0xBD 0xBD 0xBD | 0xF0 | See below | CK\_sum |

Payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte Offset | Format | Name | Scale | Unit | Drscription |
| 8 | u64 | IMEI | 1 | - | IMEI number（little-endian） |
| 2 | x16 | version |  | - | Bitfield see below |
|  |  |  |  |  |  |

This request must have a response of 0xF1, otherwise the login will fail.

例：bdbdbdbdf09b51731bc6160300000014 （imei:869465050010011）

BDBDBDBD：4-byte message header

F0：Message ID

9b51731bc6160300：imei number (Little-endian mode)，imei decimal is 869465050010011，In hexadecimal, it is0x000316C6 1B73 519B, Little-endian mode:9b51731bc6160300

0000：version

14：checksum

When TCP creates a new connection, it first reports an F0 request, which includes the IMEI. The server then records this IMEI, and all the data in this connection will be associated with this IMEI.

### 3.1.2LNK-RPL(0xF1) Connection Reply (TCP only) - Important

|  |  |
| --- | --- |
| Message | LNK-RPL |
| Description |  |
| Firmware | 　 |
| Direction | Terminal <= Terminal Server |
| Payload length | 4 bytes |
| Message structure | Header | Message ID | Payload | Checksum |
| Timestamp(unix) | 0xF1 | See below | CK\_sum |

Payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte Offset | Format | Name | Scale | Unit | Drscription |
| 4 | u32 | Token | 1 | - | Token:BDBDBDBD |

eg：28D4DE55F1BDBDBDBDEB

 The response must be replied on the current channel. The first 4 bytes are the timestamp, and the device side synchronizes the time based on this, so it must be the correct timestamp value in little-endian mode. The reply is sent byte by byte, for example, BD counts as one byte.

The timestamp refers to the number of seconds or milliseconds from the current time to 1970-1-1 0:00. Milliseconds are easier to convert in Java. Our protocol uses seconds, so it's a long integer. We use little-endian preference, so it becomes the following format: 28D4DE55 --> 55 DE D4 28 is the actual value, and then this value is converted to a long integer. Adding this value to 1970-1-1 0:00 gives the current time.

Example: Timestamp = 07FD8860 real value 6088FD07 = 1619590407 seconds. This is 1619590407 seconds from 1970-01-01 00:00:00, which is the time reported by the device: 2021/4/28 6:13:27.

### 3.1.3New heartbeat packet protocol (0xF9) - Important

After the terminal connects to the server, it uploads a heartbeat packet to the server at fixed intervals.

|  |  |
| --- | --- |
| Message | MSG\_HTB\_UPL |
| Decription | heartbeat packet |
| Firmware | -/- |
| Payload Length | 15 bytes |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0xF9 | 见下方定义 | CK\_sum |

payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte offset | Format | Name | Scale | Unit | Decription |
| 1 | U8 | Bat\_type |  |  | Battery Type0: 4-level1: 5-level2: Percentage3: Voltage |
| 2 | u16 | Bat\_volt |  | -/- | Battery LevelIf Bat\_type is 0:The battery level ranges from 0 to 3(0 for 25%, 3 for 100%)If Bat\_type is 1:The battery level ranges from 0 to 4(0 for 20%, 4 for 100%)If Bat\_type is 2:The battery level ranges from 0 to 100 |
| 1 | U8 | Signal\_type |  |  | Signal Type0: Percentage1: 5-level2: CSQ value |
| 2 | I16 | Signal\_strength |  |  | Signal\_strength |
| 1 | U8 | Other\_type |  |  | Extension Type0: Full Step Count1: Incremental Step Count2: Vibration |
| 4 | U32 | Num |  |  | Extended value |
| 4 | U32 | Timestamp | -/- | -/- | Utc Timestamp |

Heartbeat packets must be replied to, and any byte can be replied, for example 0x01. The device will consider the connection still exists as long as it receives a reply.

Reply example (can be fixed reply to this): BDBDBDBDF301

### 3.1.4Heartbeat protocol (0xF6) (used in previous devices, will not coexist with F9)

After the terminal connects to the server, it will upload a heartbeat packet to the server at regular intervals. The later versions will also include a timestamp.

|  |  |
| --- | --- |
| Message | MSG\_HTB\_UPL |
| Decription | heartbeat packet |
| Firmware | -/- |
| Payload Length | 11 bytes  |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0xF6 | See below | CK\_sum |

payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte offset | Format | Name | Scale | Unit | Decription |
| 2 | U16 | Bat\_volt |  | -/- | The number of battery power bars. |
| 4 | U32 | Step\_num |  |  | Step count data |
| 1 | U8 | Signal\_strength |  |  | Signal strength |
| 4 | U32 | Timestamp |  |  | timestamp (a timestamp will be added at the end when making up for missing data) |

eg：BDBDBDBDF6030000000000509C75FE6350

Bat\_volt=0 represents the battery percentage is 10%

 Bat\_volt=1 represents the battery percentage is 30%

 Bat\_volt=2 represents the battery percentage is 60%

 Bat\_volt=3 represents the battery percentage is 100%

## 3.2 Positioning related reporting

### 3.2.1GPS/BDS Position Reporting: Location Data Reporting (0x03)

|  |  |
| --- | --- |
| Message | MSG\_UPL\_GPS |
| Decription | 回馈GPS/BDS定位数据 |
| Firmware | -/- |
| Payload Length | 23 bytes |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0x03 | See below | CK\_sum |

payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte offset | Format | Name | Scale | Unit | Decription |
| 8 | Double | lon |  | -/- | longitude |
| 8 | Double | lat |  | 　 | latitude |
| 1 | U8 | north\_south |  |  | /\*N or S\*/ |
| 1 | U8 | east\_west |  |  | /\*E or W\*/ |
| 1 | U8 | status |  |  | /\*A or V\*/ |
| 4 | U32 | Timestamp  |  |  | Timestamp |

Example： BDBDBDBD03000000C0424C5E4000000000A5DC3C404E4541E62C616078

Lon: 000000C0424C5E40Lat: 00000000A5DC3C40 4E –N 45-E 41-A Time E62C6160

Lon:121.191574Lat: 28.861893

Status = A indicates that the information content is accurate. It can be resolved to V and can be abandoned.

Example of GPS parsing (JAVA)：

DBDBDBDB037d9f84ac81815c40e766926b1d8936404e4541749d695f0b

//DBDBDBDB03 7d9f84ac81815c40 e766926b1d893640 4e 45 41 749d695f 0b

public static void main(String[] args){

 //eg:7d9f84ac81815c40 -->405c8181ac849f7d

Double.longBitsToDouble(Long.parseLong("405c8181ac849f7d",16))); //114.02353966666665

//message e766926b1d893640 actual value 4036891d6b9266e7

Double.longBitsToDouble(Long.parseLong("4036891d6b9266e7", 16))) ;//22.535605166666667

HexToStr(data.Substring(“4e”)); //N

HexToStr(data.Substring(“45”));//E

HexToStr(data.Substring(“41”));//AA represents data "OK", V represents a warning

//Message 749d695f actual value 5f699d74

 Date date=new Date();

date.setTime(Long.parseLong(“5f699d74",16)\*1000);

SimpleDateFormatsdf = new SimpleDateFormat("yyyyMMddHHmmss");

System.out.println(sdf.format(date)); //2020-09-22 14:45:08

### 3.2.2Upload of Wi-Fi and base station information (0xA4 improved version)

|  |  |
| --- | --- |
| Message | EXT-CIU |
| Description | Cell information upload.Upload cell tower information for terminal server to compute location. |
| Firmware | 　 |
| Direction | Terminal -> Terminal Server |
| Payload length |  |
| Message structure | Header | Message ID | Payload | Checksum |
| Token | 0xA4 | See below | CK\_sum |

Payload contents:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Byte Offset** | **Format** | **Name** | **Scale** | **Unit** | **Description** |
| 4 | U32 | UtcTime |  |  | Search Time |
| 1 | u8 | Cell\_cnt | 1 | - | Number of cell info payload.Valid value:1~7 |
| 2 | u16 | Cell[0].MCC | - | - | mobile country code of cell[0] |
| 2 | u16 | Cell[0].MNC | - | - | mobile network code of cell[0] |
| 2 | u16 | Cell[0].LAC | - | - | Location area code of cell[0] |
| 4 | U32 | Cell[0].CELL\_ID | - | - | Cell id of cell[0] |
| 2 | i16 | Cell[0].RSSI | - | dbm | RSSI in dbm of cell[0] |
| … | 　 | 　 | 　 | 　 | 　 |
| 　2 | u16 | Cell[cell\_cnt-1].LAC | - | - | Location area code of cell[[cell\_cnt-1] |
| 　4 | U32 | Cell[cell\_cnt-1].CELL\_ID | - | - | Cell id of cell[[cell\_cnt-1] |
| 　2 | I16 | Cell[cell\_cnt-1].RSSI | - | dbm | RSSI in dbm of cell[[cell\_cnt-1] |
| 1 | U8 | Wifi\_cnt |  |  | Number 0f wifi |
| 1 | U8 | Wifi[0].bssid[0] |  |  |  |
| 1 | U8 | Wifi[0].bssid[1] |  |  |  |
| 1 | U8 | Wifi[0].bssid[2] |  |  |  |
| 1 | U8 | Wifi[0].bssid[3] |  |  |  |
| 1 | U8 | Wifi[0].bssid[4] |  |  |  |
| 1 | U8 | Wifi[0].bssid[5] |  |  |  |
| 4 | I32 | Wifi[0].rssi |  |  |  |
|  |  |  |  |  |  |
| 1 | U8 | Wifi[Wifi\_cnt-1].bssid[0] |  |  |  |
| 1 | U8 | Wifi[Wifi\_cnt-1].bssid[1] |  |  |  |
| 1 | U8 | Wifi[Wifi\_cnt-1].bssid[2] |  |  |  |
| 1 | U8 | Wifi[Wifi\_cnt-1].bssid[3] |  |  |  |
| 1 | U8 | Wifi[Wifi\_cnt-1].bssid[4] |  |  |  |
| 1 | U8 | Wifi[Wifi\_cnt].bssid[5] |  |  |  |
| 4 | I32 | Wifi[wifi\_cnt].rssi |  |  |  |

Note: A total of 7 base station information will be provided, including the serving cell and the adjacent 6 cells.

Message ID:A4

BDBDBDBDA468984C5F01CC010000C21871F543009E00078CBEBE1A8162C6FFFFFFC061180AF42AC1FFFFFF200BC726E000B6FFFFFFA8154DF6517EB2FFFFFFE005C5B1F824CCFFFFFFE8FCAFA02663AFFFFFFF6409805B2B9CAEFFFFFF94

 To finally obtain the latitude and longitude data, please refer to the relevant explanation in Section 4.4 of the "Explanation of WiFi positioning regarding protocol parsing for A3, A2, and A4"And amap wifi positioning protocol（It is currently only available in the Chinese version）



### 3.2.3 Bluetooth positioning information(LBE Location)（MsgId=0xD6）

|  |  |
| --- | --- |
| Message | MSG\_HTB\_UPL |
| Decription | LBE Location |
| Firmware | -/- |
| Payload Length | 1 bytes +n |
| Message structure | Hearer | Message ID | Payload | Checksum　 |
| token | 0xD6 | See below | checksum |

 Payload:

|  |  |  |  |
| --- | --- | --- | --- |
| Format | Name | Scale | Description |
| U8 | Type | 1 | Currently fixed to0 (Fix value 0) |
| U8 | Total\_groups | 1 | Total number of assemblies, there may be multiple sets of information, and each set may contain multiple items.ibeacon (The total number of groups, there may be multiple groups of information, and there may be multiple ibeacons in each group) |
| Int32 | Utc | 4 | Utctimestamp (the UTC timestamp of the first group) |
| U8 | Total\_PackCount | 1 | Total number of packages at the current time: (the ibeacon’s count of the first group) |
| U16 | Major0 | 2 | Major |
| U16 | Minor0 | 2 | Minor |
| S8 | Rssi0 | 1 | Rssi |
| U16 | MajorN | 2 | Major |
| U16 | MinorN | 2 | Minor |
| S8 | RssiN] | 1 | Rssi |
| Int32 | Utc | 4 | Utc timestamp (UTC timestamp of the second group) |
| U8 | Total\_PackCount | 1 | Total number of packages at the current time (the ibeacon’s count of current group) |
| U16 | Major0 | 2 | Major |
| U16 | Minor0 | 2 | Minor |
| S8 | Rssi0 | 1 | Rssi |
| U16 | MajorN | 2 | Major |
| U16 | MinorN | 2 | Minor |
| S8 | RssiN] | 1 | Rssi |

Example：

bdbdbdbdd60001be20315f0443271794ac43273094aa4327b956a54327fe94a56a

**bdbdbdbd - header**

**d6 - msgID**

**00 - type**

**01 -- Only one set of iBeacon data（total groups of beacons data :1）**

be20315f -- The first group’s timestamp: 0x5f3120be=1597055166

There are 4 beacon information in the first group.4327 --- major : 0x2743 = 10051

1794--- minor: 0x9417 = 37911

ac--- rssi: 0xac = -84

4327 --- major: 0x2743 = 10051

3094--- minor:0x9430 = 37936

aa--- rssi:-86

4327 --- major: 0x2743 = 10051

b956--- minor:0x56b9 = 22201

a5--- rssi:-91

4327 --- major: 0x2743 = 10051

fe94--- minor:0x94fe=38142

a5--- rssi:-91

6a --checksum

**3.3 Alarm related reporting**

### 3.3.1Alarm data upload (0x02)

|  |  |
| --- | --- |
| Message | LNK-WRN |
| Description | Terminal uploads its warnings to terminal server. |
| Firmware | 　 |
| Direction | Terminal => Terminal Server |
| Payload length | 6 bytes |
| Message structure | Header | Message ID | Payload | Checksum |
| Token | 0x02 | See below | CK\_sum |

Payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte Offset | Format | Name | Scale | Unit | Drscription |
| 2 | x16 | Upl\_warn | - | - | Bitfield see below(Little-endian) |
| 4 | U32 | Timestamp |  |  | Timestamp (timestamp will be added afterwards for supplementary upload) |
|  |  |  |  |  |  |

Bitfield WRN:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15 |  |  |  |  |  |  |  |  |  |  | 4 |  |  | 1 | 0 |

 Below is the corresponding definition table for when the bit is 1 and the current alarm. Multiple alarms may also be present simultaneously.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| bit | Name | Description | Code | 调整后的16进制 | 十进制 |
| 15 | Gas alarm（- special equipment support） | Gas alarm（- special equipment support） | 0 | 8000 |  |
| 14 |  |  | 1 | 4000 | 4\*4096=16384 |
| 13 | Fence alarm（- special equipment support） | Fence alarm (away from the beacon) （- special equipment support）） | 2 | 2000 | 2\*4096=8192 |
| 12  | Strap damage（- special equipment support） | Strap damage（- special equipment support） | 3 | 1000 | 1\*4096=4096 |
| 11 | Lock open（- special equipment support） | Lock open（- special equipment support） | 4 | 0800 | 8\*256=2048 |
| 10  | Approaching beacon point (125K) （- special equipment support） | Approaching beacon point (125K) （- special equipment support） | 5 | 0400 | 4\*256=1024 |
| 9 | Release key（- special equipment support） |  | 6 | 0200 | 2\*256=512 |
| 8 | Wearing device | Wearing device | 7 | 0100 | 1\*256=256 |
| 7 | Cancel SOS | Cancel SOS | 8 | 0080 | 8\*16=128 |
| 6 | Vibration alarm（- special equipment support） | Vibration alarm（- special equipment support） | 9 | 0040 | 4\*16=64 |
| 5 | Sedentary alarm | Sedentary alarm | 10 | 0020 | 2\*16=32 |
| 4 | Remove device | Remove device | 11 | 0010 | 1\*16=16 |
| 3 | Open box alarm（- special equipment support） | Open box alarm（- special equipment support） | 12 | 0008 | 8 |
| 2 | Shutdown | Shutdown | 13 | 0004 | 4 |
| 1 | SOS |  | 14 | 0002 | 2 |
| 0 | Low Battery | Low Battery | 15 | 0001 | 1 |

BDBDBDBD02020007FD8860E7 true value 0002—0000 0000 0000 0010 SOS alarm

Timestamp = 07FD8860 is equal to 6088FD07 = 1619590407. This represents the time when 1619590407 seconds have elapsed since 1970-01-01 00:00:00. This corresponds to the device reporting the time as 2021/4/28 6:13:27.

Alarm 02 and each independent, the status of the corresponding bit is not associated in the context of non-related alarms, only the current bit alarm with a value of 1 is processed, and the corresponding processing is performed.

###  3.3.2Alarm data upload(0x21) (supplement to 0x02)

|  |  |
| --- | --- |
| Message | LNK-WRN |
| Description | Terminal uploads its warnings to terminal server. |
| Firmware | 　 |
| Direction | Terminal => Terminal Server |
| Payload length | 8 bytes |
| Message structure | Header | Message ID | Payload | Checksum |
| Token | 0x21 | See below | CK\_sum |

Payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte Offset | Format | Name | Scale | Unit | Drscription |
| 2 | U16 | type |  |  | Type of alarm |
| 4 | U32 | Upl\_warn | - | - | Bitfield see below(Small end priority) |
| 4 | U32 | Timestamp |  |  | Timestamp (timestamp will be added afterwards for supplementary upload) |
|  |  |  |  |  |  |

Alarm type =1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 2 | Device is charging and powered off (firmware functionality support required) |  | 29 | 0004 | 4 |
| 1 | Device low on power, powered off (firmware functionality support required) |  | 30 | 0002 | 2 |
| 0 | Device powered off manually (firmware functionality support required) |  | 31 | 0001 | 1 |

BDBDBDBD21010001000000ECFFBE65DA

**3.4Equipment information and status reporting**

### 3.4.1Status parameter reporting (MSGID=0xA9) - no need for parsing

|  |  |
| --- | --- |
| Message | MSG\_HRD\_DATA |
| Decription | Terminal => Server  |
| Firmware | -/- |
| Payload Length | 6 bytes |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0xA9 | See below | CK\_sum |

1. payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte offset | Format | Name | Scale | Unit | Decription |
| 1 | u8 | TypeCnt | -/- | -/- | Type number |
|  |  |  |  |  | (Reserved 00） |
| 1 | u8 | Type | -/- | -/- | Type 1 |
| 1 | U8 | NameLen | -/- | -/- | Type1 length |
| n | N\*u8 | Name | -/- | -/- | name |
| 1 | u8 | Type | -/- | -/- | Type 2 |
| 1 | U8 | NameLen | -/- | -/- | Type 2 length |
| n | N\*u8 | Name | -/- | -/- | name |
|  |  |  |  |  |  |

Report one message upon startup

Type specified screen system (MCU module sensor wifi screen Bluetooth)

Code MCU 00 module 01 sensor 02 sequentially expands

Name lengthBDBDBDBDA9 01 00 00---mcu

125732303050475F4534322E57472E4D4C3238C8

### 3.4.2 Upload of software version and model（0XBB）- no need for parsing

|  |  |
| --- | --- |
| Message | MSG\_VERSION\_DATA |
| Decription | Upload of software version and model，Terminal=>Terminal Server |
| Firmware | -/- |
| Payload Length | 2+N bytes |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0xBB | See below | CK\_sum |

payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte offset | Format | Name | Scale | Unit | Decription |
| 1 | UINT8 | Version\_len | Required | / | The length of the software version number |
| N | S8[n] |  | Required |  | Software Version Number |
| 1 | UINT8 | Model\_len | Optional |  | Model length |
| N | S8[n] |  | Optional |  | Model Number |
| expandable |  |  |  |  | expandable |
|  |  |  |  |  |  |

Note: Previous firmware version used, now do reserved

### 3.4.3 ICCID upload of SIM card (0xF3)

|  |  |
| --- | --- |
| Message | LNK-LIN |
| Description | The terminal reports iccid to the server .when it logs on first |
| Firmware | 　 |
| Direction | Terminal => Terminal Server |
| Payload length | 10bytes |
| Message structure | Header | Message ID | Payload | Checksum |
| 0xBD 0xBD0xBD0xBD | 0xF3 | See below | CK\_sum |
| Byte Offset | Format | Name | Scale | Unit | Drscription |
| 10 | 10\*U8 | ICCID | 1 | - | ICCID number |

iccid:89861118236001639994

 message :BDBDBDBDF389861118236001639994CC

Note:Start up and report

**3.5 Downstream feedback report**

### 3.5.1Downlink feedback(MSGID=0xC0)

|  |  |
| --- | --- |
| Message | MSG\_HRD\_DATA |
| Decription | Terminal =>Server |
| Firmware | -/- |
| Payload Length | 1 +n bytes |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0xC0 | See below | CK\_sum |

payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte offset | Format | Name | Scale | Unit | Decription |
| 1 | U8 | length | -/- |  | Message ID length |
| N | n\*U8 | type | -/- |  | N Message ID |
|  |  |  |  |  |  |

This command is used for the feedback of downlink commands, returning the Message ID(s) received earlier (multiple Message ID can be returned collectively).

### 3.5.2Message status reporting(MSGID=0x28)

|  |  |
| --- | --- |
| Message | MSG\_HRD\_DATA |
| Decription | Terminal =>Server |
| Firmware | -/- |
| Payload Length | 1 bytes |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0x28 | See below | CK\_sum |

payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Bytes | Format | Name | Scale | Unit | Decription |
| 4 | U32 | timestamp | -/- | -/- | Time stamp |
| 1 | U8 | type | -/- | -/- | Type (corresponding to 0X28 downlink), currently 0x03 |
| 1 | U8 | status | -/- | -/- | Message status1: Read, received |
| 4 | U32 | serial number | -/- | -/- | (corresponding to 0X28 downlink) |
|  |  |  |  |  |  |

# **4.Setting**

## 4.1Downlink

### Set periodic upload（0x17）

The platform sets 1-4 time slots to send to the terminal. After the terminal receives the data, it uploads the data within the specified time slot.

|  |  |
| --- | --- |
| Message | MSG\_NB\_SLT |
| Decription | downlink |
| Firmware | -/- |
| Payload Length | 28 bytes  |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0x17 | See below | CK\_sum |

payload contents

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Byte offset**  |  **Format**  |  **Name**  | **Scale**  |  **Unit**  | **Decription** |  |
| 1 | u8 | 　enable | -/- | -/- | Enabled? | Time Slot 1 |
| 1 | U16 | Interval |  |  | Time interval (minutes) |
| 1 | u8 | time\_start\_h | 　 | 　 | -h |
| 1 | u8 | time\_start\_m | 　 | 　 | -m |
| 1 | u8 | time\_end\_h | 　 | 　 | -h |
| 1 | u8 | time\_end\_m | 　 | 　 | -m |
| 1 | u8 | 　enable | -/- | -/- | Enabled? | Time Slot 2 |
| 1 | U16 | Interval |  |  | Time interval (minutes) |
| 1 | u8 | time\_start\_h | 　 | 　 | -h |
| 1 | u8 | time\_start\_m | 　 | 　 | -m |
| 1 | u8 | time\_end\_h | 　 | 　 | -h |
| 1 | u8 | time\_end\_m | 　 | 　 | -m |
| 1 | u8 | 　enable | -/- | -/- | Enabled? | Time Slot 3 |
| 1 | U16 | Interval |  |  | Time interval (minutes) |
| 1 | u8 | time\_start\_h | 　 | 　 | -h |
| 1 | u8 | time\_start\_m | 　 | 　 | -m |
| 1 | u8 | time\_end\_h | 　 | 　 | -h |
| 1 | u8 | time\_end\_m | 　 | 　 | -n |
| 1 | u8 | 　enable | -/- | -/- | Enabled? | Time Slot 4 |
| 1 | U16 | Interval |  |  | Time interval (minutes) |
| 1 | u8 | time\_start\_h | 　 | 　 | -h |
| 1 | u8 | time\_start\_m | 　 | 　 | -m |
| 1 | u8 | time\_end\_h | 　 | 　 | -h |
| 1 | u8 | time\_end\_m | 　 | 　 | -m |

Example：

bd bd bd bd 17 01 03 00 00 00 13 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 dd

From 0:00 to 19:00, location is updated every 3 minutes.

BDBDBDBD17010A000000173B00000000000000000000000000000000000000000097

From 0:00 to 23:59, location is updated every 10 minutes.

### 4.1.2Message Send（MSGID=0X28）

|  |  |
| --- | --- |
| Message | MSG\_HRD\_DATA |
| Decription | Server =>Terminal downlink |
| Firmware | -/- |
| Payload Length | 7+n bytes |
| Message structure | Hearer | Message ID | Payload | Checksum |
| token | 0x28 | See below | CK\_sum |

payload contents

|  |  |  |  |
| --- | --- | --- | --- |
| Byte size | Format | Name | Decription |
| 1 | U8 | type | (Message type, if it is downlink information, the fixed value is 03) |
| 4 | Uint32 | seqID | Information id, uniqueness |
| 1 | U8 | CONTENT LEN | Contect Length |
| N | N | CONTENT | content, Chinese is GB2312 code, English is ascii code |
|  |  |  |  |
|  |  |  |  |

Example：bd bd bd bd 28 03 03 00 00 00 0b 68 65 6c 6c 6f 2c 77 6f 72 6c 64dd

 Type: 03

seqId:03 00 00 00

CONTENT LEN: 0b (10)

CONTENT :68 65 6c 6c 6f 2c 77 6f 72 6c 64 ( hello,world)

### 4.1.3Setting（0XCE）---See detailed instructions-Important Downside

|  |  |
| --- | --- |
| Message | MSG\_HRD\_DATA |
| Decription | setting，Server => Terminal downlink |
| Firmware | -/- |
| Payload Length | 4+n bytes |
| Message structure | Header | Message ID | Payload | Checksum |
| token | 0xCE | See below | CK\_sum |

payload contents

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Byte offset | Format | Name | Scale | Unit | Decription |
| 0 | u8 | Type | -/- | -/- | Type see below |
| 1 | u8 | Valid | -/- | -/- | Temporary valid, always valid |
| 2 | U16 | Len | -/- | -/- | Length of the following instruction |
| 4 | n |  | -/- | -/- | Main text |
|  |  |  |  |  |  |

Type 01 Positioning

Valid 00 Always valid 01 Effective this time 02 Off

Length Length of the following instruction

Main text:

Type

 01 Positioning function The main text can only be the basic class 01 - gps 02 - wifi 03 - Bluetooth beacon 04 - LBS base station 05 - 125k and later expanded basic class The main text can be a combination of 01 or 010204;

Such as 010203 means using gps wifi Bluetooth beacon;

 Example: wifi positioning priority (wifi> Bluetooth> gps): BDBDBDBDCE0100030002030133

gps positioning priority (gps> wifi> Bluetooth): BDBDBDBDCE0100030001020333

 Bluetooth positioning priority (Bluetooth beacon deployment required, Bluetooth> wifi> gps): BDBDBDBDCE0100030003020133,

 Analysis example: wifi positioning priority (wifi> Bluetooth> gps): BDBDBDBD CE 01 00 0300 020301 33

Type: 01; Valid: 00; length: 0300; type: 020301; checksum: 33