# Communication Protocol for GPS&OBD Tracker

**V1.09**

# Modify History Record

| No. | Version | Amended content | Date |
|---|---|---|---|
| 1 | V1.0 | Draft | 2015-12-2 |
| 2 | V1.01 | | 2016-9-10 |
| 3 | V1.02 | | 2016-11-7 |
| 4 | V1.03 | | 2016-12-3 |
| 5 | V1.04 | | 2017-4-15 |
| 6 | V1.05 | | 2017-4-25 |
| 7 | V1.06 | | 2017-7-12 |
| 8 | V1.07 | | 2017-9-8 |
| 8 | V1.08 | | 2017-9-14 |
| 9 | V1.09 | | 2017-10-7 |



## Catalogue

# Communication Protocol

## 1. Overview

This document specifies the communication protocol between the vehicle-mounted terminal and the communication gateway. The protocol sets the messages from the monitoring center to the terminal to be downlink instructions, and the messages from the terminal to the monitoring center to be uplink instructions.

# 2. Term、Definition and Breviary

## 2.1. Term and Definition

Abnormal data communication link

unregister

## 2.2. Breviary

APN——(accesspoint name)

GZIP——(GNUzip)

SMS——(shortmessageservice)

TCP——(transmissioncontrolprotocol)

TTS——(textto speech)

VSS——(vehiclespeedsensor)

# 3. Message Format

## 3.1. Communication type

By default, the terminal works in the GPRS mode, and transfers the messages between the terminal and the monitoring center through TCP/IP or UDP protocol. When getting out of the GPRS signal coverage area, the SMS text is available,

Between terminal and server, between terminal and handset(via BLE), between terminal and PC(via USB), the message format is same.

## 3.2. Definition of Data Type

Table 3.2.1 Data Type

| Data type | Description |
|---|---|
| BYTE | 1byte unsigned integral type, Range: [0, 255] |
| WORD | 2 bytes unsigned integral type, Range: [0, 65535] |
| DWORD | 4 bytes unsigned integral type, Range: [0, 4294967295] |
| BYTE[n] | n bytes unsigned integral type |
| BCD[n] | 8421 code，n bytes unsigned integral type |
| STRING | ASCII string, Variable length |
| INT8 | 1 byte signed integral type, Range: [-128, 127] |
| INT16 | 2 byte signed integral type, Range: [-32768, 32767] |
| INT32 | 4 byte signed integral type, Range: [-2147483648, 2147483647] |

| | |
|---|---|
| TIME | 7 BYTEs, GMT time, BYTE[0-1]: year, BYTE[2]: month, BYTE[3]: day, BYTE[4]: hour, BYTE[5]: minute, BYTE[6]: second<br>Example: GMT13:30:22, August 12, 2016, the data is:<br>0xe0, 0x07, 0x08, 0x0c, 0x0d, 0x1e, 0x16 |

### 3.3. Transmission regulation

If no special instruction this protocol uses little-ending mode to transfer word and double words including WORD, INT16, DWORD, INT32, and TIME in above list, agreed as follow:

**Word transmission**: transmit low 8 bits [b7, b0], then high 8 bits [b15, b8].

**Double words transmission**: transmit lowest 8 bits [b7, b0], then the lower 8 bits [b15, b8], then the higher 8 bits [b23, b16], last the highest 8 bits [b31, b24].

### 3.4. Message Format

3.4.1 Format

A complete message include five part: package header flag, message header,message body, checksum, package end flag; as follow table 3.4.1

table 3.4.1

| package header flag | message header | message body | Checksum | package end flag |
|---|---|---|---|---|
| 1 BYTE | 20 BYTEs | N BYTEs | 1 BYTE | 1 BYTE |

3.4.2 package header flag/ package end flag

package header flag and package end flag are same, 0x7e，if that has 0x7e in message header, message body, and Checksum, that should be transferred, the rule as follow:

0x7e———>0x7d 0x02

0x7d———>0x7d 0x01

example：

Original message: 0x30 0x7e 0x08 0x7d 0x55 ，Then the message package is：0x7e 0x30 0x7d 0x02 0x08 0x7d 0x01 0x55 0x7e。

3.4.3 message header

message header define as follow table3.4.3：

Table3.4.3

| location | Field | type | Field description |
|---|---|---|---|
| 0 | Message ID | WORD | 2 bytes |
| 2 | Message attribute | WORD | 2 bytes, detail as table3.4.3.1 |
| 4 | Device ID | BCD[10] | 10 bytes |
| 14 | running number | WORD | 2 bytes, loop add one by one. |

| | | | |
|---|---|---|---|
| 16 | Addition field | DWORD | 4 bytes, in Message(ID:0xD001), addition field is rental order ID, it is fixed 0 in other message. |

Message attribute as follow table3.4.3.1

table3.4.3.1

| bit15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | bit0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| reserve | | reserve | Encrypt type | | | message body length in byte | | | | | | | | | |

Encrypt type：

Bit12-bit10： =000b： no encryption,

=001b： IDEA encryption,

Other ：TBD

IDEA encryption，example：

Secret key:

0x6B,0x65,0x79,0x73,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39,0x6B,0x65,0x79,

original:

0x54,0xA9,0xBB,0xC2,0x63,0xFE,0xE0,0x09,0xB6,0x10,0xEF,0x9F,0xD5,0xF3,0x8F,0x3E,

ciphertext:

0xd9,0x65,0x3b,0xf6,0x8a,0xa8,0x76,0xc2,0x7e,0x25,0x20,0x17,0xf2,0x52,0xc6,0x22,

IDEA encryption/decryption code reference Appendix B.

### 3.4.4 checksum

Checksum for fields message heade and message body, XOR one byte by one byte.



## 4. Message List

Table4

| type | Launch | | | response | | | |
|---|---|---|---|---|---|---|---|
| | direction | Message ID | Description | direction | Message ID | Description | Apply to |
| Bsaic | up | 0x0001 | Login | down | 0x8102 | Login response | IP1/IP2 |
| | up | 0x0002 | Logout | down | 0x8101 | Server Common response | IP1/IP2 |
| | up | 0x0003 | Heartbeat | down | 0x8101 | response | IP1/IP2 |
| | down | 0x8001 | Parameters set | up | 0x0101 | Terminal Common response | IP1/IP2/BLE/USB |
| | down | 0x8002 | Parameters query | Up | 0x0102 | response | IP1/IP2/BLE/USB |
| | down | 0xFFFD | Raw command | Up | 0x7FFD | response | IP1/IP2/BLE/USB |

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| | down | 0xFFFE | Terminal Self test query | Up | 0x7FFE | response | IP1/IP2/BLE/USB |
| OBD data | up | 0x0004 | Working data | Down | 0x8101 | response | IP1/IP2/BLE/USB |
| | up | 0x0005 | DTCs data | Down | 0x8101 | response | IP1/IP2/BLE/USB |
| | up | 0x0006 | Alarm data | Down | 0x8101 | response | IP1/IP2/BLE/USB |
| | up | 0x0007 | OBD data | Down | 0x8101 | response | IP1/IP2/BLE/USB |
| | down | 0x8003 | Working data query | Up | 0x0103 | response | IP1/IP2/BLE/USB |
| | down | 0x8004 | DTCs data query | Up | 0x0104 | response | IP1/IP2/BLE/USB |
| | down | 0x8005 | Alarm data query | Up | 0x0105 | response | IP1/IP2/BLE/USB |
| | down | 0x8006 | OBD data query | Up | 0x0106 | response | IP1/IP2/BLE/USB |
| | down | 0x8007 | DTCs clear query | Up | 0x0107 | response | IP1/IP2/BLE/USB |
| Remote control | up | 0x1001 | Rental order execute result | Down | 0x8101 | response | IP1 |
| | down | 0x9001 | Remote control query | Up | 0x1101 | response | IP1/IP2/BLE/USB |
| | down | 0x9002 | Rental order query | Up | 0x1102 | response | IP1/IP2/BLE/USB |
| | down | 0x9003 | Rental order cancel query | Up | 0x1103 | response | IP1/IP2/BLE/USB |
| BLE control | down | 0xD001 | Control query from BLE | Up | 0x5101 | response | BLE |
| Remote Firmware Update | up | 0x7001 | Update query(FTP) | down | 0xF101 | response | IP1 |
| | down | 0xF201 | Update query(TCP) | Up | 0x7201 | response | IP1/IP2/BLE/USB |
| | up | 0x7202 | Fimware data query(TCP) | Down | 0xF202 | response | IP1/IP2/BLE/USB |

Note:

1、Message type have：basic message、OBD data message、remote control message、BLE control message、firmware update message；

2、Message direction：

“up” means terminal send message and receiver is server ,handset(via BLE), PC(via USB).

“Down” means server, handset(via BLE), PC(via USB) send message and receiver is terminal.

3、When receive a messag, Response message is necessary, common response is available for null response data.

4、In table4, IP1: main server, IP2:vice server, BLE: BLE equipment( phone), USB: PC. Example:

Message ID(0x8001) and Message ID(0x0101), Apply to IP1/IP2/BLE/USB, Means this message is available for terminal between mian server, vice server, BLE, and USB communication.

5、The following document that only lists the message body field, the other fields are the same, no longer.

# 5. basic message

## 5.1 uplink instruction

Sender is terminal, and receiver is server or phone or PC。

### 5.1.1 terminal common response

Message ID：0x0101。Message body detail Reference table5.1.1。

table5.1.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server, phone, PC |
| 2 | Message ID | WORD | This is the query message ID from server,phone,PC |
| 4 | result | BYTE | 0：OK；1：fail；<br>2：message mistake；3：no support |

example：

message：7E0101050000000000000000000002221B0100000000140001800 0AA7E

| data | field | Length( byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0101 | Message ID | 2 | 0x0101 |
| 0500 | Message attribute | 2 | Message body Length is 0x0005 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 1B01 | running number | 2 | 0x011b |
| 00000000 | addition field | 4 | 0x00000000 |
| 1400 | query message running number | 2 | 0x0014 |
| 0180 | query message ID | 2 | 0x8001 |
| 00 | result | 1 | 0x00 |
| aa | checksum | 1 | 0xaa |
| 7e | Package end | 1 | 0x7e |

### 5.1.2 Login

Message ID：0x0001。Message body is empty。

Terminal always sends login message first whenever TCP/UDP connection is established. It will not send other messages until receives response from server. If no response received it will keep sending login message.

example：
message：7E0100000000000000000000000222020100000000227E

| data | field | Length( byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0100 | Message ID | 2 | 0x0001 |
| 0000 | Message attribute | 2 | Message body Length is 0x0000 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 0201 | running number | 2 | 0x0102 |
| 00000000 | addition field | 4 | 0x00000000 |
| 22 | checksum | 1 | 0x22 |
| 7e | Package end | 1 | 0x7e |

### 5.1.3 Logout

Message ID：0x0002。Message body is empty。
Terminal sends logout message at trip end, then closes TCP/UDP connection and goes into sleep.

example：
message：7E0200000000000000000000000222A10100000000827E

| data | field | Length( byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0200 | Message ID | 2 | 0x0002 |
| 0000 | Message attribute | 2 | Message body Length is 0x0000 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| A101 | running number | 2 | 0x01a1 |
| 00000000 | addition field | 4 | 0x00000000 |
| 82 | checksum | 1 | 0x82 |
| 7e | Package end | 1 | 0x7e |

### 5.1.4 Heartbeat

Message ID：0x0003。Message body is empty。
In login state, terminal will send heartbeat message if there has not sended or received any data to server for more than 30 seconds, server should respond to it. Heartbeat interval can be configured by server、BLE、PCTOOL。

example：
message：7E0300000000000000000000000002229C0100000000BE7E

| data | field | Length( byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0300 | Message ID | 2 | 0x0003 |
| 0000 | Message attribute | 2 | Message body Length is 0x0000 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 9c01 | running number | 2 | 0x019c |
| 00000000 | addition field | 4 | 0x00000000 |
| be | checksum | 1 | 0xbe |
| 7e | Package end | 1 | 0x7e |

### 5.1.5 parameter query response

Message ID：0x0102。Message body detail Reference table 5.1.5。
If server sends message（ID: 0x8002），then terminal response message ID is 0x0102

Table5.1.5

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server, phone, PC |
| 2 | Number of parameters | BYTE | |
| 3 | parameters list | | Detail reference table5.2.2.2 |

example：
message：7E02010D00000000000000000000002221B00000000000D00021800022D4E2900021E00767E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0201 | Message ID | 2 | 0x0102 |
| 0d00 | Message attribute | 2 | Message body Length is 0x000d bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 1b00 | running number | 2 | 0x001b |
| 00000000 | addition field | 4 | 0x00000000 |
| 0d00 | query message running number | 2 | 0x000d |
| 02 | Number of parameters | 1 | 0x02 |
| 1800 | Parameter ID | 2 | Parameter ID=0x0018 Main server port |
| 02 | Parameter data length | 1 | Length=0x02 |
| 2d4e | Parameter data | 2 | Main server port=20013 |
| 2900 | Parameter ID | 2 | Parameter ID=0x0029 Working data updata interval time |
| 02 | Parameter data length | 1 | 0x02 |
| 1e00 | Parameter data | 2 | Working data updata interval time=30s |
| 76 | checksum | 1 | 0x76 |
| 7e | Package end | 1 | 0x7e |

### 5.1.6 raw command response

Message ID：0x7FFD。Message body is text message or other raw data。
If server sends message ( ID: 0xFFFD ), then terminal response message ID is 0x7FFD.

### 5.1.7 self test response

Message ID：0x7FFE。Message body detail Reference table5.1.6。
If server sends message ( ID: 0xFFFE ) , then terminal response message ID is 0x7FFE.

table5.1.6

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server, phone, PC |
| 2 | Self test ID | BYTE | =0: system test<br>=1：GPS test<br>=2：GSM AT test<br>=3：OBD test<br>=4：GYRO test<br>=5：device reset |
| 3 | Result data length | WORD | |
| 4 | Result | STRING | |

### 5.2 downlink instruction

Sender is server or phone or PC, and receiver is terminial。

#### 5.2.1 common response

Message ID：0x8101,  Message body  detail reference table5.2.1。

table5.2.1

| location | Field | Data | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from terminial |
| 2 | Message ID | WORD | This is the query message ID from terminial |
| 4 | result | BYTE | 0：OK；1：fail；2：message mistake；3：no support; |

example:

message:  7E0181050000000000000000000002220E00000000001C00040000B37E

| data | field | Length( byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0181 | Message ID | 2 | 0x8101 |
| 0500 | Message attribute | 2 | Message body Length is 0x0005 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 0e00 | running number | 2 | 0x000e |
| 00000000 | addition field | 4 | 0x00000000 |
| 1c00 | query message running number | 2 | 0x001c |
| 0400 | query message ID | 2 | 0x0004 |
| 00 | result | 1 | 0x00 |
| B3 | checksum | 1 | 0xb3 |
| 7e | Package end | 1 | 0x7e |

#### 5.2.2 parameters set

Message ID：0x8001。Message body  detail reference table5.2.2.1。

If server sends message ( ID: 0x8001 ) , then terminal response message ID is 0x0101.

table5.2.2.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Number of Parameters | BYTE | |
| 1 | parameters data | | Detail reference table5.2.2.2 |

example:
message:7E01802D0000000000000000000002226F0000000000313000B3132312E31352E372E3434
05F11700073133383030303030303030000000000000000000000F000101067E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0180 | Message ID | 2 | 0x8001 |
| 2d00 | Message attribute | 2 | Message body Length is 0x002d bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 6f00 | running number | 2 | 0x006f |
| 00000000 | addition field | 4 | 0x00000000 |
| 03 | Number of parameters | 1 | 0x03 |
| 1300 | Parameter ID | 2 | Parameter ID=0x0013<br>Main server IP address |
| 0b | Parameter data length | 1 | Length=0x0b |
| 3132312E31352E372E3434 | Parameter data | 11 | Main server IP address is "121.15.7.44" |
| 05f1 | Parameter ID | 2 | Parameter ID=0xf105<br>SMS phone book |
| 17 | Parameter data length | 1 | Length=0x17 |
| 00073133383030303030303030000000000000000000 | Parameter data | 23 | Book location=0x00<br>Function=0x07<br>Phone number=13800000000" |
| 0f00 | Parameter ID | 2 | Parameter ID=0x000f<br>Main server switch |
| 01 | Parameter data length | 1 | Length=0x01 |
| 01 | Parameter data | 1 | Main server switch=1 |
| 06 | checksum | 1 | 0x06 |
| 7e | Package end | 1 | 0x7e |

table5.2.2.2

| field | Data type | Description |
|---|---|---|
| Parameter ID | WORD | detail reference table5.2.2.3 |
| Length | BYTE | |
| Data | BYTE | |

table5.2.2.3 parameter detail Description

| Parameter ID | Data type | Description | permissions<br>S: set permission<br>R : read permission |
|---|---|---|---|
| 0x000F | BYTE | Main server switch， 0=close， 1=open，<br>Default: 1 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0x0010 | STRING | network APN，max length 50 bytes<br>Default: null | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0x0011 | STRING | network use name，max length 50 bytes<br>Default: null | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0x0012 | STRING | network use password，max length 19 bytes<br>Default: null | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |

<table>
<tr><td>0x0013</td><td>STRING</td><td>Main server IP address max length 50 bytes<br>Default: null</td><td>S: IP1/IP2/BLE/USB/SMS<br>R: IP1/IP2/BLE/USB/SMS</td></tr>
<tr><td>0x0018</td><td>WORD</td><td>Main server TCP port<br>Default: 0</td><td>S: IP1/IP2/BLE/USB/SMS<br>R: IP1/IP2/BLE/USB/SMS</td></tr>
<tr><td>0x0019</td><td>WORD</td><td>Heartbeat interval time，uint：second，<br>default：30，=0: no heartbeat</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0x001F</td><td>BYTE</td><td>Vice server switch， 0=close， 1=open，<br>Default: 0</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0x0023</td><td>STRING</td><td>vice server IP address max length 50 bytes<br>Default: null</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0x0028</td><td>WORD</td><td>vice server TCP port<br>Default: 0</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0x0029</td><td>WORD</td><td>Working data message(ID:0x0004) uploading interval time<br>uint：second，=0: not uploading，<br>default：120 seconds。</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0xF105</td><td>BYTE[23*N]</td><td>SMS phone book
<table>
<tr><td>location</td><td>item</td><td>type</td><td>Description</td></tr>
<tr><td>0</td><td>No</td><td>BYTE</td><td>No is [0～4]</td></tr>
<tr><td>1</td><td>function</td><td>BYTE</td><td>Bit0: receive alarm permission，<br>0：no, 1: ok<br>Bit1: parameters set and read permission<br>0：no， 1：ok<br>Bit2: control and wakeup<br>0：no， 1：ok</td></tr>
<tr><td>2</td><td>Phone</td><td>BYTE[21]</td><td>ASCII code。</td></tr>
</table></td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0xF106</td><td>BYTE[6]</td><td>SMS secret key ,ASCII code<br>Default: null</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0xF107</td><td>BYTE</td><td>SMS alarm switch<br>0: close; 1:open, default: 0</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0xF108</td><td>BYTE</td><td>SMS language<br>0: English; 1:chinese, default: 0;</td><td>S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0xF109</td><td>BCD[10]</td><td>Device ID，BCD 8421code。<br>Can not be seted</td><td>S: USB<br>R: USB</td></tr>
<tr><td>0xF10A</td><td>STRING</td><td>Product model，<br>can not be set。Max length 50 bytes。</td><td>S: none<br>R: IP1/IP2/BLE/USB</td></tr>
<tr><td>0xF10B</td><td>STRING</td><td>Firmware version，<br>can not be set。Max length 50 bytes。</td><td>S: none<br>R: IP1/IP2/BLE/USB</td></tr>
</table>

| | | | |
|---|---|---|---|
| 0xF10C | TIME | UTC time | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xF10D | WORD[2] | wakeup parameters<br>WORD[0]：wakeup interval time，<br>uint：minute，default：120 minute<br>WORD[1]：wakeup work time，<br>uint：minute，default：10 minute | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xF10E | STRING | BLE device name<br>max length 29 bytes。 | S: USB<br>R: IP1/IP2/BLE/USB |
| 0xF10F | BYTE | Encryption switch，=0：close， =1：open,<br>default：0，this switch is for terminal and server commuincation | S: IP1/USB/SMS<br>R: IP1/USB/SMS |
| 0xF110 | BYTE[16] | Secret key，default： “@bcdefghijklmno@”，<br>This switc is for terminal and server commuincation | S: IP1/USB<br>R: IP1/USB |
| 0xFFFF | BYTE | Restore factory settings<br>0x00: restore in addition to the network parameters of other parameters to the default values<br>0x01: All parameters to default values<br>0x02: Clear FLASH saved all data except parameters | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA000 | BYTE | GPS Privacy switch,<br>0：close，1：open，default：0<br>=1: GPS data is 0 in all uploaded message。 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA001 | WORD[4] | GPS data acquisition type：<br>WORD0：type，Bit0: time，Bit1: distance，Bit2: angle，default：0x0001<br>WORD1: time，uint：second， default：120<br>WORD2: distance, uint：meter， default：100<br>WORD3: angle，uint：degree， default：30 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA002 | BYTE | GPS system select：<br>Bit1: chinese（BDS） Bit2: American GPS（GP）<br>Bit3: GLONASS（GL），default：0x06 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA003 | BYTE | GPS switch in terminal wakeup，<br>0：close， 1：open，default：0 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |

| | | | |
|---|---|---|---|
| 0xA004 | BYTE[1+N*18] | Electronic fence parameters, Each electronic fence for the rectangular area<br>Data structure: fence number+first fence data+second fence data+....<br>Fence number is 1 BYTE，max 12；<br>Fence data is 18 BYTEs，structure：<br>attribute+top left Latitude+top left Longitude+low right Latitude + low right Longitude +sign<br>attribute：1 BYTE，<br>Bit6=1: out alarm，Bit5=1: in alarm<br>longitude and latitude：1 DWORD，<br>uint：0.000001 degree<br>sign：1 BYTE<br>Bit0: top left，1：east， 0：west；<br>Bit1: top left，1：north， 0：south；<br>Bit2: low right，1：east， 0：west；<br>Bit3: low right，1：north， 0：south； | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA005 | BYTE | AGPS switch，If closed, no AGPS function<br>0：open， 1：close，default：0 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA006 | STRING | AGPS vendor code，max 50 bytes， | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA007 | STRING | AGPS project code，max 50 bytes， | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA020 | BYTE | vehicle type select：<br>0：passenger car，1：heavy duty，<br>2：passenger car+ heavy duty, 3: tracker<br>Default: 2 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA021 | BYTE[2] | Fuel parameters:<br>BYTE[0]：engine capacity，uint：0.1L，<br>default：20（2.0L）<br>BYTE[1]：fuel type，=0x10 Gas =0x20 LPG =0x30 Hybrid =0x40 Diesel a =0x50 Diesel，<br>default：0x10, | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA022 | BYTE [1+N*2] | passenger car PID parameters，<br>data structure：PID number(N)+PID code list<br>PID number(N)： 1 BYTE, max 10<br>PID code： one PID is 1 WORD。<br>Default：PID number(N)=0 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA023 | BYTE [1+N*2] | Heavy duty(J1939) PID parameters，<br>data structure：PID number(N)+PID code list<br>PID number(N)： 1 BYTE, max 10<br>PID code： one PID is 1 WORD。<br>Default：PID number(N)=0 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |

| | | | |
|---|---|---|---|
| 0xA024 | BYTE<br>[1+N*2] | Heavy duty(J1708) PID parameters，<br>data structure：PID number(N)+PID code list<br>PID number(N)： 1 BYTE， max 10<br>PID code： one PID is 1 WORD。<br>Default：PID number(N)=0 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA025 | DWORD[2] | DWORD[0]：total mileage，uint：meter<br>DWORD1：total fuel，uint：0.1L | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA026 | BYTE<br>[2+N*2] | PID table supported ，can not be seted<br>structure：OBD type+PID number+PID code list<br>OBD type： 1 BYTE， 0：car OBD， 1：J1708<br>2：J1939<br>PID number： 1 BYTE,<br>PID code list： one PID is 1 WORD， | S: none<br>R: IP1/IP2/BLE/USB |
| 0xA027 | WORD | DTC detection time interval，uint：second，<br>=0：no detect，default：600 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA028 | WORD | OBD data message(ID:0x0007) uploading interval time, uint：second，<br>=0: not uploading， default：0。 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |

| | | | |
|---|---|---|---|
| 0xA040 | WORD[14] | Alarm threshold<br>WORD[0]: passenger car low valtage Alarm threshold，<br>Uint: 0.1V, default：110<br>WORD[1]: heavy duty low valtage Alarm threshold，<br>Uint: 0.1V, default：220<br>WORD[2]：Coolant temperature threshold，<br>uint：degree，default：99<br>WORD[3]：Speed threshold，<br>uint：KM/H，default：130<br>WORD[4]：rotate speed threshold，<br>uint：rpm，default：4500<br>WORD[5]：Idle engine alarm，<br>uint：minute，default：10<br>WORD[6]：fatigue driving threshold，<br>uint：minute，default：180<br>WORD[7]：Urgent to accelerate threshold，<br>uint：(1/128)g，default：51<br>WORD[8]：Sharp deceleration threshold，<br>uint：(1/128)g，default：76<br>WORD[9]：Quick lane change threshold，<br>uint：0.03degree/s，default：220<br>WORD[10]：Sharp turn threshold，<br>unit：0.03degree/s，default：830<br>WORD[11]：Collision threshold，<br>uint：(1/128)g，default：153<br>WORD[12]：Towing threshold，<br>uint：(1/128)g，default：12<br>WORD[13]：Vibration threshold，<br>uint：(1/128)g，default：4 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |

| | | | |
|---|---|---|---|
| 0xA041 | WORD[3] | Alarm switch，<br>=1：open， =0：close，<br>default: 0xffff, 0xffff, 0xfff<br>WORD[0]：<br>Bit0: power on alarm<br>Bit1: power off alarm<br>Bit2: trip start alarm<br>Bit3: trip end alarm<br>Bit4: Illegal to open the door alarm<br>Bit5: SOS alarm<br>Bit6: Timeout not security alarm<br>Bit7: Illegal fire alarm<br>Bit8: Unauthorized access alarm<br>Bit9: Low Voltage Alarm<br>Bit10: Electronic fence alarm<br>WORD[1]：<br>Bit0: MIL alarm<br>Bit1: Coolant temperature alarm<br>Bit2: speed alarm<br>Bit3: rotate speed alarm<br>Bit4: Idle engine alarm<br>Bit5: fatigue driving<br>Bit6: DTC alarm<br>Bit7: Mileage abnormal alarm<br>WORD[2]：<br>Bit0: Urgent accelerate or Sharp deceleration alarm<br>Bit1: Urgent accelerate<br>Bit2: Sharp deceleration<br>Bit3: Quick lane change alarm<br>Bit4: Sharp turn alarm<br>Bit5: Collision alarm<br>Bit6: Towing alarm<br>Bit7: Vibration alarm<br>Bit8: cartwheel alarm | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |

| | | | |
|---|---|---|---|
| 0xA060 | WORD[5] | Acc on voltage parameters，<br>WORD[0]:<br>Passenger car acc on voltage threshold，<br>uint: 0.1V， default：130<br>WORD[1]：<br>Passenger car acc on voltage offset，<br>Uint: 0.1V， default：4<br>WORD[2]：<br>Heavy duty acc on voltage threshold<br>uint: 0.1V， default：260<br>WORD[3]：<br>Heavy duty acc on voltage offset，<br>Uint: 0.1V， default：8<br>WORD[4]：<br>ACC OFF delay,<br>Uint: second，default：60 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA061 | BYTE | The voltage error is revised<br>Uint: 0.1V，default：6<br>Caused by the error of circuit parameters, the error of the voltage reading, need to modify the value. | S: USB<br>R: IP1/IP2/BLE/USB |
| 0xA080 | STRING | Vehicle VIN，read only， max 20 bytes。 | S: none<br>R: IP1/IP2/BLE/USB |
| 0xA081 | STRING | Vehicle plate，max 20 bytes。 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA082 | STRING | Vehicle model, max 20 bytes。 | S: IP1/IP2/BLE/USB<br>R: IP1/IP2/BLE/USB |
| 0xA083 | WORD | I/O parameters<br>Bit0: ACC line,<br>=0: no =1：available，default：0<br>Bit1: door status line,<br>=0: no =1：available,default：0<br>Bit2: open door control line,<br>=0: no =1：available，default：0<br>Bit3: close door control line,<br>=0: no =1：available，default：0<br>Bit4: oil control line,<br>=0: no =1：available，default：0 | S: IP1/IP2/BLE/USB/SMS<br>R: IP1/IP2/BLE/USB/SMS |
| 0xA084 | BYTE | Debug output，<br>=0: output to BLE，<br>=1：output to USB，<br>=2：no output<br>Default: 0 | S: IP1/IP2/BLE/USB/SMS<br>R: IP1/IP2/BLE/USB/SMS |

### 5.2.3 parameters read

Message ID：0x8002。Message body detail reference table5.2.3。
If server sends message ( ID: 0x8002 ) , then terminal response message ID is 0x0102.

Table5.2.3

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Number of Parameters | BYTE | |
| 1 | parameters ID list | | Detail reference table5.2.2.3 |

example:
message: 7E028007000000000000000000022287000000000003130005F10F00C97E

| data | field | Length( byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0280 | Message ID | 2 | 0x8002 |
| 0700 | Message attribute | 2 | Message body Length is 0x0007 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 8700 | running number | 2 | 0x0087 |
| 00000000 | addition field | 4 | 0x00000000 |
| 03 | Number of parameters | 1 | 0x03 |
| 130005f10f00 | Parameters ID list | 6 | 0x0013: main server IP address<br>0xf105: SMS phone book<br>0x000f: main server switch |
| C9 | checksum | 1 | 0xc9 |
| 7e | Package end | 1 | 0x7e |

### 5.2.4 Login response

Message ID：0x8102。Message body detail reference table5.2.4。
When the terminal send the login request message (0x0001), server judge the legitimacy of the terminal, if legally, use the login response (0x8102) reply.

table5.2.4

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from terminial |
| 2 | UTC time | TIME | Terminal correction according to the time. |

example:
message: 7E028109000000000000000000022266000000000AB00E007090B012C28877E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0281 | Message ID | 2 | 0x8102 |
| 0900 | Message attribute | 2 | Message body Length is 0x0009 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 6600 | running number | 2 | 0x0066 |
| 00000000 | addition field | 4 | 0x00000000 |
| Ab00 | Query message running number | 2 | 0x00ab |
| E007090B012C28 | UTC time | 7 | UTC time is: September 11, 2016, 01:44:40 |
| 87 | checksum | 1 | 0x87 |
| 7e | Package end | 1 | 0x7e |

### 5.2.5 raw command query

Message ID：0xFFFD。Message body is text command or raw command。
When the server to send the request message(0xFFFD) , then terminal response message ID is 0x7FFD.

### 5.2.6 self test query

Message ID：0xFFFE。Message body detail reference table5.2.5。
When the server to send the request message(0xFFFE) , terminal will self-test, and report the test results to server (0x7FFE).

table5.2.5

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Self test ID | BYTE | =0: system test<br>=1：GPS test<br>=2：GSM AT test<br>=3：OBD test<br>=4：GYRO test<br>=5：device reset |
| 1 | GSM AT command length | WORD | |
| 3 | GSM AT command | STRING | |

## 6.OBD data Message

### 6.1 uplink instruction

Sender is terminal, and receiver is server or phone or PC。

### 6.1.1 working data Message

Message ID：0x0004。Message body detail reference table6.1.1。
When trip on，terminial will upload working data to server on time ，Upload interval can be seted by server(parameter ID is 0x0029)。

table6.1.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Trip ID | WORD | Usually the trip ID from 1 cycle accumulation, the next trip ID is a trip ID plus 1,<br>Under trip end , the upload data of trip ID is 0 |
| 2 | time | TIME | The current terminal time |
| 9 | Total mileage | DWORD | uint：meter |
| 13 | Trip mileage | DWORD | uint：meter, under trip end, trip mileage is 0 |
| 17 | Total fuel consumption | DWORD | uint：0.1L, |
| 21 | Trip fuel consumption | DWORD | uint：0.1L，under trip end, trip fuel consumption is 0 |
| 25 | Remain fuel percentage | BYTE | uint：1%， |
| 26 | Remain power percentage | BYTE | uint：1%， |
| 27 | voltage | WORD | uint：0.1V |
| 29 | Vehicle speed | BYTE | uint：KM/H |
| 30 | rotate speed | WORD | uint：RPM |
| 32 | Coolant Temperature | INT8 | uint：degree |
| 33 | instantaneous fuel consumption | WORD | uint：0.1L/H |
| 35 | engine load | BYTE | uint：% |
| 36 | Environment temperature | INT8 | uint：degree |

| | | | |
|---|---|---|---|
| 37 | Status of vehicle | DWORD [2] | DWORD[0]:<br>Bit0-Bit1：Front left door<br>00b：OFF，01b：ON，11b：unknown<br>Bit2-Bit3：Front right door<br>00b：OFF，01b：ON，11b：unknown<br>Bit4-Bit5：rear left door<br>00b：OFF，01b：ON，11b：unknown<br>Bit6-Bit7：rear right door<br>00b：OFF，01b：ON，11b：unknown<br>Bit8-Bit9：door lock<br>00b：lock，01b：unlock，11b：unknown<br>Bit10-Bit11：anti -theft<br>00b：remove，01b：OK，10b：alarm，11b: unknown<br>Bit12-Bit13：trunk<br>00b：OFF，01b：ON，11b：unknown<br>Bit14-Bit15：engine cover<br>00b：OFF，01b：ON，11b：unknown<br>Bit16-Bit17：hand brake<br>00b：down，01b：up，11b：unknown<br>Bit18-Bit19：foot brake<br>00b：up，01b：down，11b：unknown<br>Bit20-Bit21：left turn light<br>00b：OFF，01b：ON，11b：unknown<br>Bit22-Bit23：right turn light<br>00b：OFF，01b：ON，11b：unknown<br>Bit24-Bit25：lamplet<br>00b：OFF，01b：ON，11b：unknown<br>Bit26-Bit27：headlamp<br>00b：OFF，01b：ON，11b：unknown<br>Bit28-Bit29：wiper |

| | | | |
|---|---|---|---|
| 45 | GPS | BYTE[17] | latitude： DWORD ，uint: 0.000001 degree<br>longitude：DWORD ，uint: 0.000001 degree<br>height： WORD， uint: 0.1meter<br>speed： WORD， uint: 0.1km/h<br>direction： WORD， uint: 0.1 degree<br>flag： BYTE Bit0 1—east，0—west。<br>Bit1 1—north，0—south。<br>Bit2-3 00---no 01---2D<br>11---3D<br>Bit4-7 star number<br>PDOP：WORD， uint: 0.01 |
| 62 | ACC status | BYTE | 0: ACC OFF, 1: ACC ON |
| 63 | OBD protocol | BYTE | 0x00: NONE 0x01: VPW 0x02: PWM 0x03: CAN11_500 0x04: CAN29_500 0x05: CAN11_250 0x06: CAN29_250 0x07: KWP2000 0x08: KWP2000M 0x09: ISO9141 0x0a: J1939 0x0b: J1708 |
| 64 | GSM CSQ value | BYTE | |
| 65 | Total Engine hour | DWORD | |
| 66 | | BYTE[9] | reserve。 |

example:
message:7E04004E00000000000000000000222 3101 00000000 5600 E007090A021B29 9D8A0100 782B0000 52000000 0A000000 00 00 8700 52 9108 5B 3400 2D 00 FFFFFFFFFFFFFFFF 8A205901E593C90678033A03210C9F8300 00000000000000000000000000000000 F37E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0400 | Message ID | 2 | 0x0004 |
| 4e00 | Message attribute | 2 | Message body Length is 0x004e bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 3101 | running number | 2 | 0x0131 |
| 00000000 | addition field | 4 | 0x00000000 |
| 5600 | Trip ID | 2 | 0x0056 |
| E007090A021B29 | UTC time | 7 | UTC time is: September 10, 2016, 02:27:41 |
| 9D8A0100 | Total mileage | 4 | Total mileage=101021 meters |
| 782B0000 | Trip mileage | 4 | trip mileage=11128 meters |
| 52000000 | Total fuel consumption | 4 | Total fuel=8.2L |
| 0A000000 | trip fuel consumption | 4 | trip fuel=1.0L |
| 00 | Remain fuel precentage | 1 | Remain fuel percentage=0% |
| 00 | Remain power precentage | 1 | Remain power percentage=0% |
| 8700 | voltage | 2 | Voltage=13.5V |
| 52 | Vehicle speed | 1 | Vehicle speed=82 km/h |
| 9108 | Roatate speed | 2 | Roatate speed=2193 RPM |
| 5B | Coolant temperature | 1 | Coolant temperature=91 degree |
| 3400 | Instantanenous fuel consumpion | 2 | 5.2 L/h |
| 2D | Engine load | 1 | Engine load=45% |
| 00 | Environment temperature | 1 | =0 degree |
| FFFFFFFFFFFFFFFF | Status of vehicle | 8 | |
| 8A205901E593C90678033A03210C9F8300 | GPS data | 17 | Lat=0x0159208a*0.000001=22.618250<br>Lon=0x06c993e5*0.000001=113.873893<br>Hight=0x0378*0.1=88.8 meter<br>Speed=0x033a*0.1=82.6km/h<br>Direct=0x0c21*0.1=310.5 degree<br>Flag=0x9f<br>PDOP=0x0083 |
| 00 | ACC status | 1 | ACC OFF |
| 00 | OBD protocol | 1 | none |
| 00 | CSQ | 1 | csq=0 |
| 00000000 | Total engine hour | 4 | Total engine hour=0hour |
| 00000000000000000000000000 | reserve | 9 | |
| F3 | checksum | 1 | |
| 7e | Package end | 1 | 0x7e |

### 6.1.2 vehicle DTC Message

Message ID：0x0005。Message body detail reference table6.1.2。
When trip on, terminal read DTC from vehicle on time, and then upload DTC code to server when vehicle has DTC.

table6.1.2

| location | Field | Data type | Description |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 0 | Trip ID | WORD | Usually the trip ID from 1 cycle accumulation, the next trip ID is a trip ID plus 1,<br>Under trip end the upload data of trip ID is 0 |
| 2 | time | TIME | The current terminal time |
| 9 | Vehicle type | BYTE | 0：passenger car，1：heavy duty,<br>2：Tracker（N=0） |
| 10 | DTC number(N) | BYTE | |
| 11 | DTC code<br>list | BYTE[4*N] | Every DTC is 4 BYTEs。<br>For passenger car，reference《passenger_ car_DTC _table.docx》<br>For heavy duty，reference《heavy_ duty _DTC_table.docx》<br>For passenger car, 4 BYTEs define as :<br><0x00,DTC_H,DTC_L,attr><br>Attr: DTC type，<br>0x03: stored code, 0x07: pending code.<br>DTC_H is DTC code high 8bits and DTC_L is low 8bits， bit7bit6 of DTC_H is DTC class，b7b6=00b:P, =01b:C, =10b:D, =11:U<br>example：0x00,0xc2,0x9a,0x03, means DTC code=U029A，and is stored code。<br><br>For heavy dutyr, 4 BYTEs define as :<br><DTC_H,DTC_L,attr，happens><br>Bit7 of attris DTC type，=1b: stored code, =0b: pending code。<br>Happens: DTC happen frequency<br>DTC_H is DTC code high 8bits and DTC_L is low 8bits，<br>example：0x0f,0xcf,0x00,0x00,<br>means DTC code=0x0FCF，and is pending code。 |

| | | | |
|---|---|---|---|
| 11+4*N | GPS | BYTE[17] | latitude： DWORD ，uint: 0.000001 degree<br>longitude：DWORD ，uint: 0.000001 degree<br>height： WORD， uint: 0.1meter<br>speed： WORD， uint: 0.1km/h<br>direction： WORD， uint: 0.1 degree<br>flag： BYTE Bit0 1—east，0—west。<br>Bit1 1—north，0—south。<br>Bit2-3 00--no 01--2D 11--3D<br>Bit4-7 star number<br>PDOP：WORD， uint: 0.01 |

example:

message: 7E 0500 2400 00000000000000000222 2201 00000000 5C00 E007090B020B0D 00 02 0000750700D00007 CF0C5801FC87CA061B010000100B03AF00 B87E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0500 | Message ID | 2 | 0x0005 |
| 2400 | Message attribute | 2 | Message body Length is 0x0024 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 2201 | running number | 2 | 0x0122 |
| 00000000 | addition field | 4 | 0x00000000 |
| 5c00 | Trip ID | 2 | 0x005c |
| E007090B020B0D | UTC time | 7 | UTC time is:<br>September 11, 2016, 02:11:13 |
| 00 | Vehicle type | 1 | 00: passenger car |
| 02 | DTC number | 1 | Have 2 DTCs |
| 0000750700D00007 | DTC list | 8 | 1th DTC code=00007507<br>2th DTC code=00D00007 |
| CF0C5801FC87CA061B010000100B03AF00 | GPS data | 17 | |
| B8 | checksum | 1 | |
| 7e | Package end | 1 | 0x7e |

### 6.1.3 alarm Message

Message ID：0x0006。Message body detail reference table6.1.3。

When trip on, there are alarms, terminal will upload alarm to server.

table6.1.3

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Trip ID | WORD | Usually the trip ID from 1 cycle accumulation, the next trip ID is a trip ID plus 1,<br>Under trip end the upload data of trip ID is 0 |
| 2 | time | TIME | The current terminal time |

| | | | |
|---|---|---|---|
| 9 | Alarm statistics | WORD[28] | statistical occurrences of alarm in history。<br>WORD[0]: power on alarm number<br>WORD[1]: power off alarm number<br>WORD[2]: trip start alarm number<br>WORD[3]: trip end alarm number<br>WORD[4]: Illegal to open the door alarm number<br>WORD[5]: SOS alarm number<br>WORD[6]: Timeout not security alarm number<br>WORD[7]: Illegal fire alarm number<br>WORD[8]: Unauthorized access alarm number<br>WORD[9]: Low Voltage Alarm number<br>WORD[10]: Electronic fence alarm number<br>WORD[11]: MIL alarm number<br>WORD[12]: Coolant temperature alarm number<br>WORD[13]: speed alarm number<br>WORD[14]: rotate speed alarm number<br>WORD[15]: Idle engine alarm number<br>WORD[16]: fatigue driving number<br>WORD[17]: DTC alarm number<br>WORD[18]: Mileage abnormal alarm number<br>WORD[19]: Urgent accelerate or Sharp deceleration alarm number<br>WORD[20]: Urgent accelerate number<br>WORD[21]: Sharp deceleration number<br>WORD[22]: Quick lane change alarm number<br>WORD[23]: Sharp turn alarm number<br>WORD[24]: Collision alarm number<br>WORD[25]: Towing alarm number<br>WORD[26]: Vibration alarm number<br>WORD[27]: cartwheel alarm number |

| | | | |
|---|---|---|---|
| 65 | Alarn happen flag | WORD[3] | WORD[0]：<br>Bit0: power on alarm<br>Bit1: power off alarm<br>Bit2: trip start alarm<br>Bit3: trip end alarm<br>Bit4: Illegal to open the door alarm<br>Bit5: SOS alarm<br>Bit6: Timeout not security alarm<br>Bit7: Illegal fire alarm<br>Bit8: Unauthorized access alarm<br>Bit9: Low Voltage Alarm<br>Bit10: Electronic fence alarm<br>WORD[1]：<br>Bit0: MIL alarm<br>Bit1: Coolant temperature alarm<br>Bit2: speed alarm<br>Bit3: rotate speed alarm<br>Bit4: Idle engine alarm<br>Bit5: fatigue driving<br>Bit6: DTC alarm<br>Bit7: Mileage abnormal alarm<br>WORD[2]：<br>Bit0: Urgent accelerate or Sharp deceleration alarm<br>Bit1: Urgent accelerate<br>Bit2: Sharp deceleration<br>Bit3: Quick lane change alarm<br>Bit4: Sharp turn alarm<br>Bit5: Collision alarm<br>Bit6: Towing alarm<br>Bit7: Vibration alarm<br>Bit8: cartwheel alarm |
| 71 | GPS | BYTE[17] | latitude： DWORD ，uint: 0.000001 degree<br>longitude：DWORD ，uint: 0.000001 degree<br>height： WORD， uint: 0.1meter<br>speed： WORD， uint: 0.1km/h<br>direction： WORD， uint: 0.1 degree<br>flag： BYTE Bit0 1—east，0—west。<br>Bit1 1—north，0—south。<br>Bit2-3 00--no 01--2D 11--3D<br>Bit4-7 star number<br>PDOP：WORD， uint: 0.01 |

Example:
Example:7E 0600 5800 00000000000000000222 2401 00000000 5C00 E007090B020B13
13000C00590043000000000000000000000059000000010000000000000000000000000000006B002E0001000000120
0000000000E000000 000001000000CF0C5801FC87CA061B010000100B03AF007C7E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0600 | Message ID | 2 | 0x0006 |
| 5800 | Message attribute | 2 | Message body Length is 0x0058 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 2401 | running number | 2 | 0x0124 |
| 00000000 | addition field | 4 | 0x00000000 |
| 5c00 | Trip ID | 2 | 0x005c |
| E007090B020B13 | UTC time | 7 | UTC time is: September 11, 2016, 02:11:19 |
| 13000C00590043000000000000000000000059000000010000000000000000000000000000006B002E0001000000120000000000E000000 | Alarm statics | 56 | Numbers of power on=0x0013<br>Numbers of power on=0x000c<br>Numbers of trip start=0x0059<br>..... |
| 000001000000 | Alarm flag | 6 | MIL alarm |
| CF0C5801FC87CA061B010000100B03AF00 | GPS data | 17 | |
| 7c | checksum | 1 | |
| 7e | Package end | 1 | 0x7e |

### 6.1.4 OBD data Message

Message ID：0x0007。Message body detail reference table6.1.4。
When trip on，terminial will upload OBD data to server on time ，Upload interval can be seted by server(parameter ID is 0xA028)。

table6.1.4

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Trip ID | WORD | Usually the trip ID from 1 cycle accumulation, the next trip ID is a trip ID plus 1,<br>Under trip end the upload data of trip ID is 0 |
| 2 | Time | TIME | The current terminal time |
| 9 | PID number | BYTE | |
| 10 | First PID code | WORD | |
| 12 | First PID data | BYTE[4] | Every PID data is 4 BYTEs，PID data detail as<br>For passenger car:<br>《passenger_car_ PID_table.docx》，<br>For heavy duty:<br>《heavy_ duty_ PID_table.docx》， |

| | | | |
|---|---|---|---|
| 16 | Second PID code | WORD | |
| 18 | Second PID data | BYTE[4] | |
| ... | | | |
| 10+6*N | GPS | BYTE[17] | latitude： DWORD ，uint: 0.000001 degree<br>longitude：DWORD ，uint: 0.000001 degree<br>height： WORD， uint: 0.1meter<br>speed： WORD， uint: 0.1km/h<br>direction： WORD， uint: 0.1 degree<br>flag： BYTE Bit0 1—east，0—west。<br>Bit1 1—north，0—south。<br>Bit2-3 00--no 01--2D 11--3D<br>Bit4-7 star number<br>PDOP：WORD， uint: 0.01 |

Example :

7E 0700 5100 88990000000000000000 D900 00000000 4D00 E0070C1401222E 09 10211AB500000521D80000000B21000000000C21490D00000F21D80000000D21520000003121780000000421470000002F2100000000000000000000000000000000000000000000857E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0700 | Message ID | 2 | 0x0007 |
| 5100 | Message attribute | 2 | Message body Length is 0x0051 bytes, encrypt type=no encrypt |
| 88990000000000000000 | Device ID | 10 | 88990000000000000000 |
| D900 | running number | 2 | 0x00d9 |
| 00000000 | addition field | 4 | 0x00000000 |
| 4D00 | Trip ID | 2 | 0x004D |
| E0070C1401222E | UTC time | 7 | UTC time is:<br>December 20, 2016, 01:34:46 |
| 09 | Number of PID | 1 | 9 PIDs |
| 10211AB500000521D80000000B21000000000C21490D00000F21D80000000D21520000003121780000000421470000002F2100000000 | PID data | 54 | PID0: 10211AB50000<br>PID code=0x2110,<br>PID data=0xb51A*0.01=463.62g/s<br><br>PID1: 0521D8000000<br>PID code=0x2105,<br>PID data=0xd8=-40 degree<br>..... |
| 00000000000000000000000000000000000 | GPS data | 17 | |
| 85 | checksum | 1 | |
| 7e | Package end | 1 | 0x7e |

### 6.1.5 Response of working data query

Message ID：0x0103。Message body detail reference table6.1.5。
When server query working data（Message ID:0x8003），terminal respone message ID is 0x0103.

table6.1.5

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | Working data | BYTE[X] | This is same as message body of 0x0004 |

Example :
7E 0301 5000 00000000000000000222 8901 00000000 EB00 5C00 E007090B021E36 EDCA0200 5E6C0000 8C000000 00000000 00 00 8D00 50 B803 00 0000 35 00 FFFFFFFFFFFFFFFF CF0C5801FC87CA061B010000100B03AF00 00000000000000000000000000000000CA7E

| data | field | Length(byte) | Data decode |
|---|---|---|---|
| 7E | Package header | 1 | 0x7e |
| 0301 | Message ID | 2 | 0x0103 |
| 5000 | Message attribute | 2 | Message body Length is 0x0050 bytes, encrypt type=no encrypt |
| 00000000000000000222 | Device ID | 10 | 00000000000000000222 |
| 8901 | running number | 2 | 0x0189 |
| 00000000 | addition field | 4 | 0x00000000 |
| EB00 | Query message running number | 2 | |
| 5c00 | Trip ID | 2 | 0x005c |
| E007090B021E36 | UTC time | 7 | |
| EDCA0200 | Total mileage | 4 | |
| 5E6C0000 | Trip mileage | 4 | |
| 8C000000 | Total fuel consumption | 4 | |
| 00000000 | trip fuel consumption | 4 | |
| 00 | Remain fuel precentage | 1 | |
| 00 | Remain power precentage | 1 | |
| 8D00 | voltage | 2 | |
| 50 | Vehicle speed | 1 | |
| B803 | Roatate speed | 2 | |
| 00 | Coolant temperature | 1 | |
| 0000 | Instantanenous fuel consumpion | 2 | |
| 35 | Engine load | 1 | |
| 00 | Environment temperature | 1 | |
| FFFFFFFFFFFFFFFF | Status of vehicle | 8 | |
| CF0C5801FC87CA061B010000100B03AF00 | GPS data | 17 | |
| 00000000000000000000000000000000 | reserve | 16 | |
| CA | checksum | 1 | |
| 7e | Package end | 1 | 0x7e |

### 6.1.6 Response of vehicle DTC query

Message ID：0x0104。Message body detail reference table6.1.6。
When server query vehicle DTC ( Message ID:0x8004 ) , terminal respone message ID is 0x0104.

table6.1.6

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | Vehicle DTC | BYTE[X] | This is same as message body of 0x0005 |
| example:<br>7E040126000000000000000000000222AE010000000006015C00E007090B022434000200007507 00D00007CF0C5801FC87CA061B010000100B03AF00277E | | | |

### 6.1.7 Response of alarm query

Message ID：0x0105。Message body detail reference table6.1.7。
When server query alarm ( Message ID:0x8005 ) , terminal respone message ID is 0x0105.

table6.1.7

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | alarm | BYTE[X] | This is same as message body of 0x0006 |
| example:<br>7E05015A00000000000000000000222C1010000000017015C00E007090B02272E13000C0059004300000000000000000000005900000001000000000000000000000000000006B002E000100000012000000000000E000000000001000000CF0C5801FC87CA061B010000100B03AF009E7E | | | |

### 6.1.8 Response of OBD data query

Message ID：0x0106。Message body detail reference table6.1.8。
When server query alarm ( Message ID:0x8006 ) , terminal respone message ID is 0x0106.

table6.1.8

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | OBD data | BYTE[X] | This is same as message body of 0x0007 |

| |
|---|
| example:<br>7E06015300889900000000000000000EC00000000007B054D00E0070C1401230509102<br>10AB500000521D80000000B21000000000C214D0D00000F21D80000000D215200000<br>03121820000000421470000002F2100000000000000000000000000000000000000000<br>0087E |

### 6.1.9 Response of DTC clear query

Message ID：0x0107。Message body detail reference table6.1.9。
When server query clear DTC ( Message ID:0x8007 ) , terminal respone message ID is 0x0107.

table6.1.9

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | result | BYTE | 0：OK， 1：fail， 2：no finish。 |
| example:<br>7E070103000000000000000000000222DA0100000000280100D77E | | | |

## 6.2 downlink instruction

### 6.2.1 working data query

Message ID：0x8003。Message body is null。
When server query workingf data ( Message ID:0x8003 ) , terminal respone message ID is 0x0103.
example:
Message: 7E038000000000000000000000000222030000000000A07E

### 6.2.2 vehicle DTC query

Message ID：0x8004。Message body is null。
When server query vehicle DTC ( Message ID:0x8004 ) , terminal respone message ID is 0x0104.
example:
Message: 7E048000000000000000000000000222190000000000BD7E

### 6.2.3 alarm query

Message ID：0x8005。Message body is null。

When server query alarm ( Message ID:0x8005 ) , terminal respone message ID is 0x0105.
example:
Message: 7E058000000000000000000000022229000000000008C7E

### 6.2.4 OBD data query

Message ID：0x8006。Message body is null。
When server query OBD data ( Message ID:0x8006 ) , terminal respone message ID is 0x0106.
example:
Message: 7E068000000000000000000000022233000000000957E

### 6.2.5 DTC clear query

Message ID：0x8007。Message body is null。
When server query DTC clear ( Message ID:0x8007 ) , terminal respone message ID is 0x0107.
example:
Message: 7E078000000000000000000000022238000000009F7E

## 7. remote control

### 7.1 uplink instruction

#### 7.1.1 report the result of order execute

Message ID：0x1001。Message body detail reference table7.1.1。
When the terminal receives the mobile phone via bluetooth or other order of near field communication equipment request command, terminal check legality of this order, and perform the corresponding action, and report result to server.

table7.1.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Time | TIME | The current terminal time |
| 7 | Order ID | DWORD | |
| 11 | User ID | DWORD | |
| 15 | Control type | BYTE | 0：open door<br>1：close door |

| | | | |
|---|---|---|---|
| 16 | result | BYTE | 0：success，1：fail，<br>2：no support this control。 |

**7.1.2 response of remote control**

Message ID：0x1101。Message body detail reference table7.1.2。
When server query remote control（Message ID:0x9001），terminal respone message ID is 0x1101.

table7.1.2

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | Control type | BYTE | |
| 3 | result | BYTE | 0：success，1：fail，<br>2：no support this control。 |

**7.1.3 response of order**

Message ID：0x1102。Message body detail reference table7.1.3。
When server query remote control（Message ID:0x9002），terminal respone message ID is 0x1102.

table7.1.3

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | Time | TIME | The current terminal time |
| 9 | Order ID | DWORD | |
| 13 | User ID | DWORD | |
| 17 | Start time | TIME | |
| 24 | End time | TIME | |
| 31 | result | BYTE | 0：success，1：fail， |

**7.1.4 response of cancel order**

Message ID：0x1103。Message body detail reference table7.1.4。
When server query remote control（Message ID:0x9003），terminal respone message ID is 0x1103.

table7.1.4

| location | Field | Data type | Description |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from server |
| 2 | Time | TIME | The current terminal time |
| 9 | Order ID | DWORD | |
| 13 | User ID | DWORD | |
| 17 | result | BYTE | 0：success，1：fail， |

**7.2 downlink instruction**

**7.2.1 remote control query**

Message ID：0x9001。Message body detail reference table7.2.1。
When server query remote control（Message ID:0x9001），terminal respone message ID is 0x1101.

table7.2.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Time | TIME | The current server UTC time |
| 7 | Control type | BYTE | =0：open door<br>=1：close door<br>=2：Disconnect the oil<br>=3：Connect the oil |

**7.2.2 order query**

Message ID：0x9002。Message body detail reference table7.2.2。
When server query order（Message ID:0x9002），terminal respone message ID is 0x1102.

table7.2.2

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Time | TIME | The current server UTC time |
| 7 | Order ID | DWORD | |
| 11 | User ID | DWORD | |
| 15 | Start time | TIME | |
| 22 | End time | TIME | |

| | | | |
|---|---|---|---|
| 29 | secret key | BYTE[16] | Adopting the IDEA encryption, for this order of mobile phone bluetooth communications with terminal data encryption |

### 7.2.3 cancel order query

Message ID：0x9003。Message body detail reference table7.2.3。
When server query cancel order ( Message ID:0x9003 ) , terminal respone message ID is 0x1103.

table7.2.3

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Time | TIME | The current server UTC time |
| 7 | Order ID | DWORD | |
| 11 | User ID | DWORD | |

## 8. BLE Instruction

### 8.1 uplink instruction

### 8.1.1 response the result of control

Message ID：0x5101。Message body detail reference table8.1.1。
When mobile phone via BLE query control ( Message ID:0xD001 ) , terminal response message ID is 0x5101.

table8.1.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Time | TIME | The current server UTC time |
| 7 | Order ID | DWORD | |
| 11 | User ID | DWORD | |
| 15 | Control type | BYTE | 0：open door<br>1：close door<br>=..... |
| 16 | result | BYTE | 0：success，1：fail，<br>2：no support this control。 |

### 8.2 uplink instruction

#### 8.2.1 query control

Message ID：0xD001。Message body detail reference table8.2.1。
When mobile phone via BLE query control ( Message ID:0xD001 ) , terminal response message ID 0x5101 to mobile phone and message ID 0x1001 to server.

table8.2.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Time | TIME | The current server UTC time |
| 7 | Order ID | DWORD | |
| 11 | User ID | DWORD | |
| 15 | Control type | BYTE | 0：open door<br>1：close door<br>=..... |

### 9. Firmware update

there are two ways to upgrade Firmware, FTP and TCP/IP.

FTP upgrade method: request initiated by the terminal, the terminal can upload upgrade at any time request (Message ID 0 x7001), decide whether to need to upgrade by server, at the same time issued request response to the terminal (Message ID 0 xf101), terminal receiving server allows response , can upgrade from FTP server at any free time.

TCP/IP upgrade methods: request initiated by the server, after the connection is established with the terminal , the server can be issued to upgrade request (request ID 0 xf201), by the terminal to determine whether to need to upgrade, at the same time upload request response to server (response ID 0 x7201), if need to upgrade, the terminal will take the initiative to send data request packet (0 x7202) to server, server receiving request packet, distributed packet (0 xf202) to the terminal. This upgrade method is also suitable for PC(via USB) and bluetooth device firmware upgrades to the terminal.

### 9.1 FTP update

#### 9.1.1 query update

Message ID：0x7001。Message body detail reference table9.1.1。
When terminal query update ( Message ID:0x7001 ) , server respone message ID is 0xF101.

table9.1.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Firmware NO | BYTE | Fixed =0 |
| 1 | product model | STRING[50] | Length=50 |
| 51 | Firmware version | STRING[50] | Length=50 |

**9.1.2 response update**

Message ID：0xF101。Message body detail reference table9.1.2。
When terminal query update ( Message ID:0x7001 ) , server respone message ID is 0xF101.

table9.1.2

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from terminal |
| 2 | Firmware NO | BYTE | Fixed =0 |
| 3 | Whether to upgrade | BYTE | =0：not,<br>=1：need |
| 4 | product model | STRING[50] | |
| 54 | Firmware version | STRING[50] | |
| 104 | FTP server IP | STRING[50] | |
| 154 | FTP erver port | WORD | |
| 156 | User name | STRING[50] | |
| 206 | password | STRING[50] | |
| 256 | path | STRING[50] | |
| 306 | File name | STRING[50] | |
| | | | |

**9.2 TCP/IP update**

**9.2.1 query update**

Message ID：0xF201。Message body detail reference table9.2.1。
When server query update ( Message ID:0xF201 ) , terminal respone message ID is 0x7201.

table9.2.1

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Update ID | WORD | Used to identify this upgrade. |
| 2 | FirmwareNO | BYTE | Fixed=0 |
| 3 | product model | STRING[50] | Length=50 |
| 53 | Firmware version | STRING[50] | Length=50 |

**9.2.2 response update**

Message ID：0x7201。Message body detail reference table9.2.2。

table9.2.2

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from serverl |
| 2 | Update ID | WORD | Used to identify this upgrade. |
| 4 | FirmwareNO | BYTE | Fixed=0 |
| 5 | Whether to upgrade | BYTE | =0：not,<br>=1：need |

**9.2.3 query firmware data**

Message ID：0x7202。Message body detail reference table9.2.3。
When terminal query firmware data（Message ID:0x7202），server respone message ID is 0xF202.

table9.2.3

| location | Field | Data type | Description |
|---|---|---|---|
| 0 | Update ID | WORD | Used to identify this upgrade. |
| 2 | FirmwareNO | BYTE | Fixed=0 |
| 3 | Package no | WORD | First package no=0;<br>One package data is 512 BYTEs |

**9.2.4 response firmware data**

Message ID：0xF202。Message body detail reference table9.2.4。

table9.2.4

| location | Field | Data type | Description |
|---|---|---|---|

| | | | |
|---|---|---|---|
| 0 | running number | WORD | This is the query message running number from terminal |
| 2 | Update ID | WORD | Used to identify this upgrade. |
| 4 | Firmware No | BYTE | Fixed=0 |
| 5 | Package no | WORD | First package no=0;<br>One package data is 512 BYTEs |
| 7 | Package flag | BYTE | =0：not，=1： is the last package |
| 8 | Data length | WORD | Length is fixed 512, beside the last package |
| 10 | Package data checksum | BYTE | Checksum for package data, XOR one byte by one byte。 |
| 11 | Package data | BYTE[X] | |



## 10. SMS instructions

## 11. Appendix A

（1） passenger car PID table：passenger_car_PID_table.docx

（2） passenger car DTC table：passenger_car_PID_table.docx

（3） heavy duty PID table：heavy_duty_PID_table.docx

（4） heavy duty DTC table：heavy_duty_DTC_table.docx

## 12. Appendix B

IDEA encryption/decryption source code：

```
#define IDEAKEYSIZE             16
#define IDEABLOCKSIZE           8
#define ROUNDS                  8
#define KEYLEN                  (6*ROUNDS+4)
#define low16(x)                        ((x) & 0xffff)
typedef uint16_t IDEAkey[KEYLEN];

static IDEAkey z,dz,T;

static uint16_t inv(uint16_t x)
{
        uint16_t t0,t1;
        uint16_t q,y;

        if (x<=1)  return x;

        t1=(uint16_t)(0x10001l/x);
        y=(uint16_t)(0x10001l%x);

        if (y==1) return low16(1-t1);

        t0=1;
        do{
                q=x/y;
                x=x%y;
                t0+=q*t1;
                if (x==1) return t0;
                q=y/x;
                y=y%x;
                t1+=q*t0;
        } while (y!=1);

        return low16(1-t1);
}

static uint16_t mul(uint16_t a, uint16_t b)
{
        uint32_t p;
```

```
        if (a)
        {
                if (b)
                {
                        p=(uint32_t)a*b;
                        b=(uint16_t)(low16(p));
                        a=(uint16_t)(p>>16);
                        return b-a+(b<a);
                }
                else
                {
                        return 1-a;
                }
        }
        else
                return 1-b;
}

#define MUL(x,y) (x=mul(low16(x),y))

static void en_key_idea(uint16_t *userkey, uint16_t *Z)
{
        int i,j;

        for (j=0;j<8;j++) Z[j]=*userkey++;

        for (i=0;j<KEYLEN;j++)
        {
                i++;
                Z[i+7]=((Z[i&7] << 9) | (Z[i+1 & 7] >> 7));
                Z+=i&8;
                i&=7;
        }
}

static void de_key_idea(IDEAkey Z,IDEAkey DK)
{
        int j;
        uint16_t t1,t2,t3;
        uint16_t *p=T+KEYLEN;

        t1=inv(*Z++);
        t2=-*Z++;
        t3=-*Z++;
        *--p=inv(*Z++);
```

```
        *--p=t3;
        *--p=t2;
        *--p=t1;

        for (j=1;j<ROUNDS;j++)
        {
                t1=*Z++;
                *--p=*Z++;
                *--p=t1;
                t1=inv(*Z++);
                t2=-*Z++;
                t3=-*Z++;
                *--p=inv(*Z++);
                *--p=t2;
                *--p=t3;
                *--p=t1;
        }

        t1=*Z++;
        *--p=*Z++;
        *--p=t1;
        t1=inv(*Z++);
        t2=-*Z++;
        t3=-*Z++;
        *--p=inv(*Z++);
        *--p=t3;
        *--p=t2;
        *--p=t1;

        for(j=0,p=T;j<KEYLEN;j++)
        {
                *DK++=*p;
                *p++=0;
        }
}



static void cipher_idea(uint16_t in[4],uint16_t out[4],IDEAkey Z)
{
        uint16_t x1,x2,x3,x4,t1,t2;
        int r=ROUNDS;

        x1=*in++; x2=*in++;
```

```
        x3=*in++; x4=*in;

        do {
                MUL(x1,*Z++);
                x2+=*Z++;
                x3+=*Z++;
                MUL(x4,*Z++);
                t2=x1^x3;
                MUL(t2,*Z++);
                t1=t2+(x2^x4);
                MUL(t1,*Z++);
                t2=t1+t2;
                x1^=t1;
                x4^=t2;
                t2^=x2;
                x2=x3^t1;
                x3=t2;
        } while (--r);

        MUL(x1,*Z++);
        *out++=x1;
        *out++=(x3+*Z++);
        *out++=(x2+*Z++);
        MUL(x4,*Z);
        *out=x4;
}

/*
*********************************************************************************************
* function:  uint16_t IDEA_Encrypt(uint16_t len, uint8_t *data, uint8_t key[16] )
*  IDEA encryption
* parameters :  @ len  : Length of the encrypted data ( Must be 8 times )
*               @ data : The encrypted data ,
*               @ key  : secret key
* result: =0 :  OK
*         =1 :  FAULT
*********************************************************************************************/
uint8_t IDEA_Encrypt(uint16_t len, uint8_t *data, uint8_t key[16] )
{
        uint16_t in[4], out[4];
        uint16_t i;
        uint16_t *p;

        if( (len%8)!=0||len==0 ) return 1;
```

```
	en_key_idea((uint16_t *)key,z);

	p=(uint16_t *)data;

	for(i=0;i<(len/8);i++){
		in[0]=p[i*4+0];
		in[1]=p[i*4+1];
		in[2]=p[i*4+2];
		in[3]=p[i*4+3];
		cipher_idea(in,out,z);
		p[i*4+0]=out[0];
		p[i*4+1]=out[1];
		p[i*4+2]=out[2];
		p[i*4+3]=out[3];
	}

	return 0;
}

/*
*********************************************************************************************
*function:  uint16_t IDEA_ Decrypt (uint16_t len, uint8_t *data, uint8_t key[16] )
*  IDEA decryption
* parameters :  @ len  : Length of the decrypted data ( Must be 8 times )
*                  @ data : The decrypted data ,
*                  @ key  : secret key
* result: =0 :  OK
*          =1 :  FAULT
*********************************************************************************************
*/
uint8_t IDEA_Decrypt(uint16_t len, uint8_t *data, uint8_t key[16] )
{
	uint16_t in[4], out[4];
	uint16_t i;
	uint16_t *p;

	if( (len%8)!=0||len==0 ) return 1;

	en_key_idea((uint16_t *)key,z);
	de_key_idea(z,dz);

	p=(uint16_t *)data;
```

```
	for(i=0;i<(len/8);i++){
		in[0]=p[i*4+0];
		in[1]=p[i*4+1];
		in[2]=p[i*4+2];
		in[3]=p[i*4+3];
		cipher_idea(in,out,dz);
		p[i*4+0]=out[0];
		p[i*4+1]=out[1];
		p[i*4+2]=out[2];
		p[i*4+3]=out[3];
	}

	return 0;
}
```