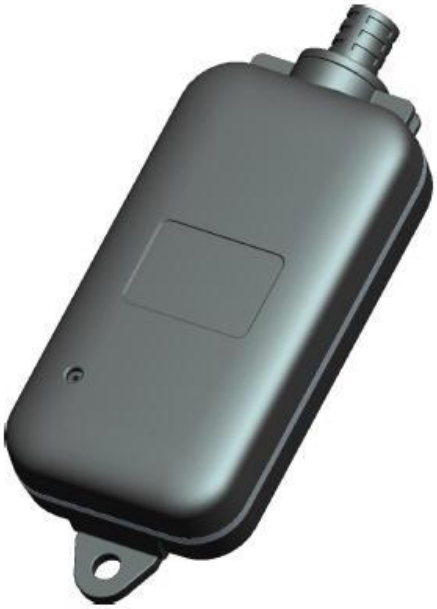
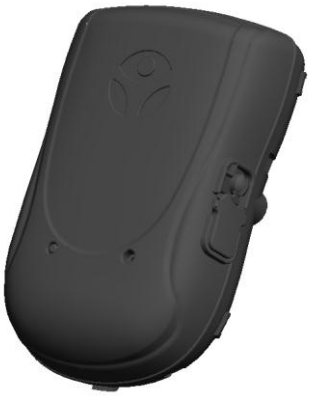
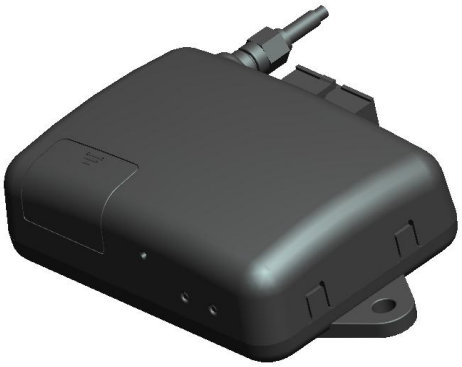
MXT-1XX Protocol Standard







Revision History

Date	Version	Description	Author
02/JUN/2009	2.0.0	Initial Version	Gustavo
04/JUN/2009	2.0.1	Add position sample and some commands.	Fei Xie; Kurt Wang
10/JUN/2009	2.0.2	1. Modify max speed configuration to UINT8 2. Modify protocol segment to UINT8 in keep alive packets	Kurt Wang
16/JUL/2009	2.0.3	1.Add anti-theft configurations 2.Add configuration for ignition off operate 3.Add buzzer control command 4.Add position sending reason to position packet	Kurt Wang
23/JUL/2009	2.0.4	Add more reasons	Kurt Wang
30/JUL/2009	2.0.5	Add configuration for enable/disable microphone Add configuration for trigger alarm when detect moving	Kurt Wang
28/AUG/2009	2.0.6	Modify the firmware update end command	Kurt Wang
17/SEP/2009	2.0.7	1. Modify the offset for each firmware 2. add some configuration for debounce timer 3. add some configuration for output mask 4. Add configurations and commands for getting position from log. 5. add “delivery fail” reason’s position.	Kurt Wang
27/OCT/2009	2.0.8	1. Add new sub-command to get all configuration 2. Add detail information about event enable/disable setting command.	Kurt Wang
10/NOV/2009	2.0.9	Add configuration for reset or power off after trigger backdoor	Kurt Wang
7/DEC/2009	2.1.0	Add some configuration Change low_power to tampering_sensor in flag	Kurt Wang
31/DEC/2009	2.1.1	Add new configuration for odometer and RPM input	Kurt Wang
14/JAN/2010	2.1.2	Add configuration for new g-sensor feature	Kurt Wang
10/FEB/2010	2.1.3	Add new configuration for new feature	Kurt Wang
10/MAR/2010	2.1.4	Add new configuration for call feature	Kurt Wang
19/MAR/2010	2.1.5	Add keep working timer configuration	Kurt Wang
26/MAR/2010	2.1.6	Add parameters range and projects accepted table	Kurt Wang
2/APR/2010	2.1.7	Add configuration for enable or disable firmware download by ZIGBEE Add driver ID group to position packet	Kurt Wang

18/MAY/2010	2.1.8	Add new configuration and new accessories protocol	Kurt Wang
18/MAY/2010	2.19	Modify the accessory configuration and list management part	Fei Xie
3/JUN/2010	2.2.0	Add ignition code configuration	Kurt Wang
18/JUN/2010	2.2.1	Add WT400 accessory information Modify load log command	Kurt Wang
24/JUN/2010	2.2.2	Add configuration for panic key and door detect as input	Kurt Wang
9/JUL/2010	2.2.3	Add configuration for GPS filter	Kurt Wang
26/JUL/2010	2.2.4	Add Jamming detection events	Kurt Wang
6/AUG/2010	2.2.5	Add output3 control to waypoint action	Kurt Wang
6/SEP/2010	2.2.6	Add new configuration for RPM	Kurt Wang
26/SEP/2010	2.2.7	Add GPS Failure configuration and Get ICCID command	Kurt Wang
22/OCT/2010	2.2.8	Add accessories text send, set protocol	Kurt Wang
5/NOV/2010	2.2.9	Add new configuration for output Add RPM to position packets	Kurt Wang
18/NOV/2010	2.3.0	Add new configuration for distance trigger position Secondary ANP, IP priority, packets encrypt key…	Kurt Wang
31/DEC/2010	2.3.1	Add Transparent Transmission command	Kurt Wang
31/JAN/2011	2.3.2	Add new configuration for g-sensor debounce	Kurt Wang
28/FEB/2011	2.3.3	Add configuration for rolling, side, shock exceed threshold output Add parking mode output configuration Add GPS speed configuration	Kurt Wang
9/MAR/2011	2.3.4	Add configuration for growing order sending old position	Kurt Wang
14/APR/2011	2.3.5	Add delete all accessories text command	Kurt Wang
13/MAY/2011	2.3.6	Add new position reason	Kurt Wang
9/JUN/2011	2.3.7	Describing CMD (0x3E)	Kurt Wang
17/JUN/2011	2.3.8	Add MXT14X and Sunbird DD value	Kurt Wang
15/JUL/2011	2.3.9	Add AGPS command	Kurt Wang
17/AUG/2011	2.4.0	Add new virtual accessories information	Kurt Wang
28/SEP/2011	2.4.1	Add new event for WT200 link broken and expand input changed	Kurt Wang
14/OCT/2011	2.4.2	Add new parameter for tag link fail output and panic button via input1 or wireless button	Kurt Wang
12/MAR/2012	2.4.3	Add new configurations	Kurt Wang
27/JUL/2012	2.4.4	Add new configurations	Kurt Wang
24/SEP/2012	2.4.5	Modify some words	Kurt Wang
27/NOV/2012	2.4.6	Add new waypoint protocol	Kurt Wang
21/MAY/2013	2.4.7	Add new event for G100	Kurt Wang

24/MAY/2013	2.4.8	Add command to calibrate ignition voltage	Kurt Wang
30/MAY/2013	2.4.9	Add Activate Panic command	Kurt Wang
14/JUN/2013	2.5.0	Add Accessories get and set command	Kurt Wang
2/SEP/2013	2.5.1	Add new configuration for stop interval factor	Kurt Wang
6/DEC/2013	2.5.2	Add RS232 transmission	Kurt Wang
9/DEC/2013	2.5.3	Add new example of position packet	Marcus Filho
3/APR/2014	2.5.4	Add new configurations	Kurt Wang
22/SEP/2014	2.5.5	Add new configurations and events	Kurt Wang
21/NOV/2014	2.5.6	Change event 50 description, modify hourmeter range	Kurt Wang
26/DEC/2014	2.5.7	Add new configurations	Kurt Wang
27/JAN/2015	2.5.8	Add Engine Seal	Kurt Wang
12/MAR/2015	2.5.9	Included new Device Descriptors (0xAB and 0xAC)	Marcus Filho
04/APR/2015	2.6.0	Included command examples; added MT: 0x50	Marcus Filho
15/MAY/2015	2.6.1	Add AGPS and LBS configuration and command	Kurt Wang
18/MAY/2015	2.6.2	Explain Protocols 0x18, 0x1A and New Data Features	Rogerio Souza
21/MAY/2015	2.6.3	Modify some information about projects	Angélica Lima
01/JUNE/2015	2.6.4	Add new commands and structure of New Data	Everton Silva
10/JUL/2015	2.6.5	Add new configurations	Kurt Wang
13/JUL/2015	2.6.6	Add new commands and configurations	Everton Silva
17/JUL/2015	2.6.7	Add new commands and configurations	Everton Silva
25/JUL/2015	2.6.8	Add new commands of CAN telemetry	Everton Silva
25/JUL/2015	2.6.8	Add new commands and update the document	Everton Silva

CONFIDENTIALITY DECLARATION

Summary

1. INTRODUCTION 7

2. GPRS/USB COMMUNICATION PROTOCOL 7

2.1 FORMAT 7
2.2 BYTE STUFFING 8

3. STANDARD COMMAND TYPES 8

3.1 ACK 8
3.2 NACK 9
3.3 POSITION PACKETS 9
3.3.1 POSITION PACKET FORMAT 9
3.3.2 ACCESSORIES INFORMATION 15
3.3.3 REPLY FROM SERVER 20
3.3.4 EXAMPLES 20
3.3.5 COMMAND FROM SERVER 25
3.4 READ CONFIGURATION 27
3.4.1 REPLY OF CRM_DEV_INFO 27
3.4.2 REPLY OF CRM_NET_ATTRIB 28
3.4.3 REPLY OF CRM_IP_ADDRESS 29
3.4.4 REPLY OF CRM_REPORT_INTERVAL 29
3.4.5 REPLY OF CRM_GSR 30
3.4.6 REPLY OF CRM_GPS 31
3.4.7 REPLY OF CRM_SMS 32
3.4.8 REPLY OF CRM_ZIG_INFO 33
3.4.9 REPLY OF CRM_OTHERS 33
3.4.10 REPLY OF CRM_PANIC_MODE 36
3.4.11 REPLY OF CRM_ZIG_INFO_EXT 37
3.4.12 REPLY OF CRM_ALL_CFG 38
3.4.13 REPLY OF CRM_NET_ATTRIB2 55
3.5 SET CONFIGURATION 56
3.6 POWER OFF 110
3.7 RESET 110
3.8 GPRS PAUSE 110
3.9 GPRS RESUME 111

3.10 WAYPOINT **111**
3.11 FIRMWARE DOWNLOAD START **114**
3.12 FIRMWARE DOWNLOAD FILE **114**
3.13 FIRMWARE DOWNLOAD END **115**
3.14 2.4GHZ ACCESSORY OPERATION **116**
3.15 KEEP ALIVE PACKET **122**
3.16 GET OLD POSITION PACKETS **123**
3.17 GET ICCID **125**
3.18 TRANSPARENT TRANSMISSION **125**
3.19 TRANSMISSION FILE START **125**
3.20 TRANSMISSION FILE DATA **126**
3.21 TRANSMISSION FILE END **126**
3.22 VERSION PACKETS **126**
3.23 NEW WAYPOINT **127**
3.24 TRANSPARENT TRANSMISSION **132**
3.24.1 DATA RECEIVE FROM RS232 133
3.24.2 CONFIGURATIONS: 134
3.25 GET DEVICE SUPPORTED PARAMETER **135**
3.26 GET DEVICE GENERAL INFORMATION **161**
3.27 NEW FEATURES DATA **163**
3.27.1 NEW FEATURES DATA ID TYPES 164

1. Introduction

This document defines the communication protocol used for development involving MXT device through GPRS/USB communication. All standard command types are listed along with mandatory or optional parameters.

2. GPRS/USB Communication Protocol

2.1 Format

Communication protocol frame format:

SOF	DD	MT	DEVID	DATA	CRC	EOF

NOTE 1: Gray fields are coded using byte stuffing method, described on next topic.

NOTE 2: Unless there are any explicit remarks, all values use “Little Endian” order (lowest byte or bit first).

SOF: Start of Frame, **0x01**, 1 byte. This byte starts every packet sent;

DD: Device descriptor, according to the following table:

If no encrypt:

0xA0: MXT-100 (portable, no 2.4Ghz transceiver)
0xA1: MXT-101 (portable, 2.4Ghz enabled)
0xA2: MXT-150 (automotive, no 2.4Ghz transceiver)
0xA3: MXT-151 (automotive, 2.4Ghz enabled)
0xA4: MXT-120
0xA5: iMXT
0xA6: MXT-140
0xA7: MXT-141
0xA8: MXT-100N
0xA9: MXT-101N
0xAA: G-100
0xAB: MXT-130
0xAC: MXT-142
0xAD: MXT-130D

For encrypted packets: The DD will add 0x10

MESSAGE TYPE: Message type, 1 byte, described more ahead in this document.
DEVID: Device ID, 4 bytes;
For encrypted packets: the high 4 bit of the high byte is using to indicate the encrypt data added bytes, because the encrypt data must be 16 bytes align, so if the original data is not, it need added some bytes to align.
DATA: Reserved for data, size-varying according with message type and destination device type. This field is coded by the byte stuffing method explained on the next topic;
CRC: The CRC16-CCITT calculation from DD until the last byte of DATA. CRC is calculated before the byte stuffing process (please notice that previous versions of MXT protocol used Big Endian for CRC);
EOF: End of Frame, **0x04**, 1 byte. This byte ends every packet sent.

2.2 Byte stuffing

The used byte stuffing method on this protocol is the insertion of one token byte (**0x10**), followed by the real byte added to 0x20. The bytes coded by this method are only: **0x01**, **0x04, 0x10, 0x11** and **0x13**.

3. Standard Command Types

The following standard frames derive from the basic structures presented on topic 2. The specific field values are presented below:

3.1 Ack

This is the general acknowledgment command. Used for confirmation of position packets and other packets when necessary.

MT: 0x02;
DATA FIELD: 2 bytes;
UINT16 CRC_received;

3.2 Nack

This is the general command reporting error. Used for reporting any error of other commands when necessary.

MT: 0x03;
DATA FIELD: 3 bytes;

UINT8 Error_type;
UINT16 CRC_received;

Error Type descriptions:

0x48 – CRC error (NACK_CRC)
0x49 – Command number error (NACK_CMD_NUM)
0x50 – Command parameter error (NACK_CMD_PAR)
0x51 – Offset error (NACK_OFFSET)
0x52 – Command execution error (NACK_ERRO_EXEC_CMD)
0x53 – Busy error (NACK_BUSY)
0x54 – Image not found error (NACK_IMG_NOT_FOUND)
0x55 – IO Module communication failure (NACK_ERRO_HCS)

3.3 Position Packets

3.3.1 Position packet format

MT: 0x31;
DATA FIELD: variable;

```
typedef struct
{
    u8    protocol;                  // 0x08 – MXT Standard Package
                                     // 0x0A – MXT Standard Package + Transparent
                                    Transmission
                                     // 0x18 – MXT Standard Package + New Features Data
                                     // 0x1A – MXT Standard Package + Transparent
                                    Transmission + New Features Data
```

u8	info_groups;	//1: enable waypoint ID (u32), wpt_group u16), accelerometer event (u8) and accelerometer value (u8) info (please see data defined as below) //1: enable wireless accessory packet transmission (8 bytes long each, quantity defined by input_mask/acc_count) //1: enable GPS_SVN (u8), GPS_HDOP (u8), GPS_SNR (u8), GSM_CSQ (u8), life after reset (u16 - minutes), input voltage (u8 – volts/5) and internal temperature (s8 – degrees) //1: enable odometer (u32 - meters) //1: enable hourmeter (u32 – minutes with ignition on) //1: enable position sending reason (u32) //1: detail information of the voltage.(u16 volts like nn.nnn),(u16 minutes the backup battery have used) //1: driver ID(4 bytes)
u16	position_count;	// auto-increment from 0 to 65535
t32_date_time	date_time;	
s32	latitude;	//Ex: -19.923293 (*1000000) > -19923293
s32	longitude;	//Ex: -19.923293 (*1000000) > -19923293
t32_flags	flags;	
u8	speed;	// km/h
u8	input_mask/acc_count	//1: input 1 (0: not masked, 1: masked)_bit0 //1: input 2 (0: not masked, 1: masked)_bit1 //1: input 3 (0: not masked, 1: masked) _bit2 //1: input 4 (0: not masked, 1: masked) _bit3 //4: qty of 8 byte long wireless packets included at the end of position packet (maximum of 15 packets) _bit4:7

} MXT1XX_POS_ITEM;

Typedef struct

Typedef struct

NOTE 1: Each info group, when present, will add its corresponding amount of bytes to packet size. They will be found immediately after input_mask/acc_count, in the same order as flags are represented inside info_groups byte (default 0x06).

NOTE 2: Cell ID information, if set in corresponding flag of t32_flags, will add another 8 bytes, immediately after last info_group present, containing Country Code (u16), Network Code (u16),

Local Area Code (u16) and Cell ID (u16).

NOTE 3: If quantity of wireless packets indicated by highest 4 bits of input_mask/acc_count is higher than zero AND corresponding flag of info_groups is set, 8 bytes will be added for each wireless packet present, according to protocol described more ahead in this document. If number of wireless packets is higher than zero, but flag for wireless packets is not set inside info_groups byte, data of wireless packets will not be added to position.

NOTE 4: the position packets sending reason define as below:
1 - Device power on
2 - GPRS first attached or reattached
3 - Transmission interval stopped
4 - Transmission interval moving
5 - Transmission interval in panic
6 - Some configuration changed (change transmission interval or modify position packet content)
7 - Server's requirement
8 - Get GPS valid after transmission interval (on transmission interval the GPS does not fix)
9 - Ignition on
10 - Ignition off
11 - Panic activated
12 - Panic deactivated
13 - Input 1 activated
14 - Input 1 opened
15 - Input 2 activated
16 - Input 2 opened
17 - Input 3 activated
18 - Input 3 opened
19 - Input 4 activated
20 - Input 4 opened
21 - Moving detect
22 - Stopped detect
23 - Anti-theft alarmed
24 - At least one accessories critical
25 - External power fail
26 - External power ok
27 - GPS antenna fail
28 - GPS antenna OK
29 - 2.4Ghz packet received
30 - Entering sleep
31 - Output 1 activated
32 - Output 1 deactivated

33 - Output 2 activated
34 - Output 2 deactivated
35 - Output 3 activated
36 - Output 3 deactivated
37 - Maximum speed exceeded
38 - Maximum speed OK (after a exceed event)
39 - Entering waypoint
40 - Leaving waypoint
41 - Backup battery fail
42 - Backup battery OK (after fail event)
43 - Delivery fail
44 - Require from SMS
45 - Tampering is open
46 - G-sensor rolling threshold reached
47 - G-sensor side threshold reached
48 - G-sensor shock threshold reached
49 - GPS direction changed
50 - SMS interval (without GPRS connection)
51 - Power off
52 - Anti-theft enter normal from alarmed
53 - GSM Jamming switches from NO to YES
54 - GSM Jamming switches from YES to NO
55 - Excessive RPM
56 - Excessive RPM on neutral
57 - Speeding on neutral
58 - GPS Failure
59 - Distance attached
60 - Power Fail and GPS Fail
61 - AGPS requires, in this case, the 8~16bits of the reason bytes using to indicate how long minutes does not get the fixed GPS data.
62 - TAG accessories status changed from 1 to 0
63 - TAG accessories battery status changed
64 - Link broken
65 - Expand input changed
66 - TAG accessories status changed from 0 to 1
67 - Only have 30% power in battery
68 - Only have 20% power in battery
69 - Keep stopped with ignition on status
70 - Improper moving
71 - Camera blind
72 - Camera blind recover

73 - Camera video lost
74 - Camera video ok
75 - RS232 data incoming
76 - Calibrate ignition voltage finished
77 - Before enter deep sleep
78 - Exceed max speed in raining
79 - Resume speed after exceed in raining
80 - Acceleration exceed
81 - Acceleration resume after exceed
82 - Deceleration exceed
83 - Deceleration resume after exceed
84 - RFID driver login
85 - RFID driver logout
86 - RFID passenger login
87 - Generic exceed max speed
88 - Generic resume speed after exceed
89 - Fail try device password more than 3 times
90 - Receive engine seal activate command
91 - Engine seal activated
92 - Engine seal deactivated
93 - Engine seal activated by relay
94 - Engine seal deactivated relay
95 - Engine seal activated by input1
96 - Engine seal deactivated by input1
97 - Network scan response
98 - Histogram of Speed
99 - Delta of Journey
100 - Telemetry events
101 - Event's reconstruction (reserved)
102 - Route's reconstruction

```
typedef union
{
    u32 value;
    struct
    {
        u32 seconds         :6;
        u32 minutes         :6;
        u32 hours           :5;
        u32 days            :15; // from 01/01/2000
    } info;
```

```
} t32_date_time;

typedef union
{
    u32 value;
    struct
    {
        u32 ignition                :1; // (0x00 off, 0x01 on)
        u32 panic                   :1; //(0x00 no panic status, 0x01 panic status)
        u32 input1                  :1; //(0x00 open, 0x01 low)
        u32 input2                  :1; //(0x00 open, 0x01 low)
        u32 input3                  :2; //(0x00 open, 0x01 low, 0x02 high)
        u32 input4                  :1; //(0x00 open, 0x01 high)
        u32 output1                 :1; //(0x00 open, 0x01 grounded)
        u32 output2                 :1; //(0x00 open, 0x01 grounded)
        u32 output3                 :1; //(0x00 open, 0x01 high)
        u32 direction               :3; //(0x00 north, 0x01 north east, 0x02 east, 0x03 south east,
                                        0x04 south, 0x05 south west, 0x06 west, 0x07 north west)
        u32 gprs_connection         :1; //(0x00 no server connection, 0x01 connected to server)
        u32 voice_cal               :1; //(0x00 no voice call, 0x01 voice call)
        u32 gps_fix                 :1; //(0x00 valid position retrieved from memory, 0x01 valid
                                        position actual)
        u32 gps_antenna_fail        :1; //(0x00 antenna OK, 0x01 antenna fail)
        u32 max_speed_exceeded      :1; //(0x00 speed OK, 0x01 max speed exceeded)
        u32 tampering_seosor        :1; //(0x00 tampering is OK, 0x01 tampering is open)
        u32 sleep                   :1; //(0x00 GPS not sleeping, 0x01 will enter sleep and turn
                                        off GPS)
        u32 battery_charging        :1; //(0x00 not charging, 0x01 charging)
        u32 battery_fault           :1; //(0x00 internal battery OK, 0x01 battery not found)
        u32 power_fail              :1; //(0x00 power ok, 0x01 no external power)
        u32 waypoint_restricted     :1; //(0x00 off, 0x01 on)
        u32 waypoint_entering       :1; //(entering waypoint)
        u32 waypoint_leaving        :1; //(leaving waypoint)
        u32 GSM_jamming             :1; //(0x00 no jamming, 0x01 jamming)
        u32 anti_theft_status       :2; //(0x00 disarmed, 0x01 armed, 0x02 suspended , 0x03
                                        alarmed)
        u32 accessory_missing       :1; //at least one expected accessory not found
        u32 moving_status           :1; //(0x00 stopped, 0x01 moving)
        u32 cell_id_present         :1; //(CC, NC, LC, ID - 8 bytes–will be present)
    } info;
} t32_flags;
```

Accelerometer data:

Accelerometer event byte:

- 4 less significant bits - show highest absolute value registered in any axis (shock, 1~8G in 0.5G range) since last position packet;
- 4 more significant bits – flags for threshold limit reached:
  - 0x80 – side threshold reached
  - 0x40 – rolling threshold reached
  - 0x20 – shock threshold reached
  - 0x10 - reserved

Accelerometer value byte:

4 less significant bits – rolling axis G average for last 4 samples (2 seconds) in G's/10 (0~1.6G range);

4 more significant bits – side axis G average for last 4 samples in G's/10; (0~1.6G range)

3.3.2 Accessories information

If wireless packets from accessories are present inside position packet, their data must be decoded according to the following protocol.

Each packet has same length (8 bytes). The structure is as below:

ID	4 Bytes
Device type	1 Byte
Protocol	4 bits (lower)
RSSI	4 bits (higher)
Data	2 Bytes

Device types actually defined are:

0x20:	WT100 (wireless watch)
0x21:	WT110 (wireless button/tag)
0x22:	WT111 (wireless temperature sensor)
0x23:	WT112 (wireless switch sensor)
0x24:	WT200 (wireless anti-theft/relay)
0x25:	WT300 (wireless LCD)
0x26:	WT400 (wireless device)
0xA1:	MXT101+ (wireless relay)
0xA3:	MXT151+ (wireless anti-theft/relay)

0xF0	RPM virtual accessory
0xF1	GPS SPEED virtual accessory
0xF2	WT110 TAG status
0xF3	WT110 TAG battery status
0xF4	Expander IO
0xF5	RFID Driver
0xF6	RFID Passenger
0xF7	SPEED Details

Data can have the following formats based on *Device type*:

0x20 - WT100 – wireless watch:

Battery level:	3bits;	//percentage of current voltage (0x00<12.5%, 0x07>87.5%).
Button status:	2bits;	//00: no pressed, 01: short pressed, 10: long pressed, 11: short pressed 3 times in 5 seconds.
Touch sensor status:	1bit;	//0: normal, 1: no body.
Wrist loop 1 event:	1bit;	//0: normal, 1: open.
Wrist loop 2 event:	1bit;	//0: normal, 1: open.
Reserved:	1bit.	
Internal temperature:	7bits;	//-40~+87 (0x00=-40 or lower, 1 degree per unit, 0x7F=+87 or higher).

0x21 - WT110 – wireless button/tag:

Battery level:	3bits	//percentage of current voltage (0x00<12.5%, 0x07>87.5%).
Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed, 11: short pressed 3 times in 5 seconds
Reserved:	4bits.	
Internal temperature:	7bits	//-40~+87 (0x00=-40 or lower, 1 degree per unit, 0x7F=+87 or higher)

0x22 - WT111 – wireless temperature sensor:

Battery level:	3bits	//percentage of current voltage (0x00<12.5%, 0x07>87.5%)
Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed, 11: short pressed 3 times in 5 seconds
Under-temperature:	1bit	//0: normal, 1: bellow -30.
Over-temperature:	1bit	//0: normal, 1: above +70
Reserved:	1bit	
External temperature:	8bits	//-128~127

0x23 - WT112 – wireless switch sensor:

Battery level:	3bits	//percentage of current voltage (0x00<12.5%, 0x07>87.5%)
Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed, 11: short

		pressed 3 times in 5 seconds
Loop 1:	1bit	//0: closed, 1: opened
Loop 2:	1bit	//0: closed, 1: opened
Reserved:	2bits	
Internal temperature:	7bits	//-40~+87 (0x00=-40 or lower, 1 degree per unit, 0x7F=+87 or higher)

0x24 - WT200 – wireless anti-theft/relay:

Ignition status:	1bit	//0: off, 1: on
Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed,
Doors status:	1bit	//0: closed, 1: opened
Mode:	3bits	//000: normal, 001: parking, 010: armed, 011: violated antitheft, 100: sleep
High-side output:	2bits	//00: not activated, 01: activated, 10: open load, 11: over-temperature/current
Low-side output 1:	2bits	//00: not activated, 01: activated, 10: reserved, 11: over-temperature/current
Low-side output 2:	2bits	//00: not activated, 01: activated, 10: reserved, 11: over-temperature/current
Under-temperature:	1bit	//0: normal, 1: bellow -30
Over-temperature:	1bit	//0: normal, 1: above +70
Reserved:	1bit	

0x25 – WT300 – wireless LCD:

OK Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed, 11: short pressed 3 times in 5 seconds
Under-temperature:	1bit	//0: normal, 1: bellow -30
Over-temperature:	1bit	//0: normal, 1: above +70
Event code:	2bits	//00: no event, 01: accept received message, 02: reject received message, 03: send pre-configured message)
Message group:	2bits	//(group of message sent)
Message code:	8bits	//(code of message accepted, rejected or sent)

0x26 – WT400 – wireless device:

External temperature:	8bits	//-128~127
Battery level:	3bits	//percentage of current voltage (0x00<12.5%, 0x07>87.5%)
Moving status	1bit	//0 is stopped, 1 is moving
Doors status:	1bit	//0: closed, 1: opened
Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed, 11: short pressed 3 times in 5 seconds
Power failed:	1bit	//0: normal, 1: external power failed

0xA1 – MXT101+ – wireless relay

GPRS indicator:	1bit	//0: No GPRS, 1: GPRS connected
Battery level:	3bits	//percentage of current voltage (0x00<12.5%, 0x07>87.5%)
Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed,
Dummy:	2bits	
Internal temperature:	8bits	//-128~127

0xA3 – MXT151+ – wireless anti-theft/relay

Ignition status:	1bit	//0: off, 1: on
Button status:	2bits	//00: no pressed, 01: short pressed, 10: long pressed,
Doors status:	1bit	//0: closed, 1: opened
Mode:	3bits	//000: normal, 001: parking, 010: armed, 011: violated antitheft, 100: sleep
High-side output:	2bits	//00: not activated, 01: activated, 10: open load, 11: over-temperature/current
Low-side output 1:	2bits	//00: not activated, 01: activated, 10: reserved, 11: over-temperature/current
Low-side output 2:	2bits	//00: not activated, 01: activated, 10: reserved, 11: over-temperature/current
Under-temperature:	1bit	//0: normal, 1: bellow -30
Over-temperature:	1bit	//0: normal, 1: above +70
GPRS indicator:	1bit	//0: No GPRS, 1: GPRS connected

NOTE: when Mode is not '011' AND GPRS indicator is '1', it will mean that it is sending broadcast due to a request via remote command.

0xF0 - RPM virtual accessory

This is only for MXT151+, to add RPM to packets. The data is the RPM value. And the ID will be 0.

0xF1 - GPS SPEED virtual accessory

This is only for MXT151+, to add GPS speed to packets. The data is the GPS speed value. And the ID will be 0.

0xF2 – WT110 TAG status

Bits in the byte 0,1,2,3,5,6 as a table indicate 48 tag devices status, 0 is absent,1 is present

0xF3 – WT110 TAG battery status

Bits in the byte 0,1,2,3,5,6 as a table indicate 48 tag devices battery status, 0 is low power, 1 is OK.

0xF4 – WT200 Expander IO

1 Byte	//Index of the WT200 in link list
2 Bytes	//I/O status (Please look this information below)
1 Byte	//Reserved
1 Byte	//Device Type (0xF4)
1 Byte	//Reserved
1 Byte	//One Wire data: Temperature sensor 1
1 Byte	//One Wire data: Temperature sensor 2

I/O Status

DATA: 2 BYTE

Input A:	1bit	//0: OFF, 1: ON
Input B:	1bit	//0: OFF, 1: ON
Input C:	1bit	//0: OFF, 1: ON
Input D:	1bit	//0: OFF, 1: ON
Mask Input A:	1bit	//0: OFF, 1: ON
Mask Input B:	1bit	//0: OFF, 1: ON
Mask Input C:	1bit	//0: OFF, 1: ON
Mask Input D:	1bit	//0: OFF, 1: ON
Output A:	2bits	//00: not activated, 01: activated, 10: open load, 11: over-temperature/current
Output B:	2bits	//00: not activated, 01: activated, 10: open load, 11: over-temperature/current
Output C:	2bits	//00: not activated, 01: activated, 10: open load, 11: over-temperature/current
G-Sensor Status:	1bit	//0: Stopped, 1: Moving
Reserved:	1bit	

0xF5 – RFID Driver

4 Bytes	//RFID driver ID
1 Byte	//Device Type (0xF5)
1 Byte	//RFID device
2 Bytes	//dummy

0xF6 – RFID Passenger

4 Bytes	//RFID passenger ID
1 Byte	//Device Type (0xF6)
1 Byte	//In list or not
2 Bytes	//dummy

0xF7 – SPEED Details

1 Bytes	// Average Speed
1 Bytes	// Max Speed
2 Bytes	// Over Speed time
1 Byte	//Device Type (0xF7)
3 Bytes	//dummy

3.3.3 Reply from server

When MXT-1xx sends a position packet to server, server must send an ACK to device in order to confirm that packet was received successfully.

3.3.4 Examples

Position packet with no present group and no cell information:

Hex data:		ASCII
31	position indication MT	
08	protocol	8
00	info_group	No group info added
CD 24	position_count	9421
DD 37 87 19	date_time	2008-12-11 19:31:29
10 19 62 02	latitude	39.983376
A0 4B EF 06	longitude	116.345760
02 AC 18 00	flags	panic set gprs connected, Direction is south east Gps fixed, sleep set, Charging,
00	speed	0
02	input_mask/acc_count	input 2 is masked and no accessoires

Position packet with default group and cell information:

Hex data:		ASCII
31	position indication MT	
08	protocol	8
06	info_group	6
9C 24	position_count	9372
BA F1 84 19	date_time	2008-12-10 15:06:58

10 19 62 02	latitude	39.983376
A0 4B EF 06	longitude	116.345760
02 AC 18 80	flags	panic set gprs connected, Direction is south east Gps fixed, sleep set, Charging, Cell id present
00	speed	0
20	input_mask/acc_count	no inputs masked and have 2
	accessoires	
01 00 00 00	accessory 1 ID	1
23	device type	WT111
20	protocol	
07 1A	Data	full battery and no button Pressed, and temperature is 26
02 00 00 00	accessory 2 ID	2
25	device type	WT200
20	protocol	
01 08	Data	ignition on and in normal Mode, button not pressed Door is closed, low side 2 is activated.

Position Packet with Wireless Info and Accelerometer info:

01a3313b11500008ff4012999ba0272c8ccffe8c555efd01a014c000400200000001008293612330000a1d10c2518591500a7d10101612330

00a1e10c2501000000f20000000a012d194a0000287e1a0000090a000004000000342c0000758b0000f427500077026309cd0004

01 - Start of frame

a3 - Device descriptor

0xa3 = 163 = MXT151+

31 - Message type

0x31 = Position packet

3b115000 - Device ID

0050113b = 5247291

08 - Protocol

ff - Info group

1 = Waypoint and Accelerometer Information

1 = Wireless accessory packet transmission

1 = General Information

1 = Odometer

1 = Hourmeter

1 = Reason of sent packets

1 = Voltage detail information

1 = Driver ID

4012 - Position count

1240 = 4672

999ba027 - Date/Time

27a09b99 = 00100111101000001001101110011001

001001111010000 = 5072 + 01/01/2000 = nov/20/2013

01001 = 09h

101110 = 46m

011001 = 25s

2c8ccffe - Latitude

fecf8c2c = 11111110110011111000110000101100

two's complement (NOT + 1) = 00000001001100000111001111010100 = -19.952596

8c555efd - Longitude

fd5e558c = 11111101010111100101010110001100

two's complement (NOT + 1) = 00000010101000011010101001110100 = -44.149364

01a014c0 - Flags

01 = 00000001

1 = Ignition = On

0 = Panic = No

0 = Input 1 = Open

0 = Input 2 = Open

00 = Input 3 = Open

0 = Input 4 = Open

0 = Output 1 = Open

a0 - 10100000

0 = Output 2 = Open

0 = Output 3 = Open

000 = Direction = North

1 = GPRS connection = Yes

0 = Voice call = No

1 = GPS fix = Yes

14 = 00010100

0 = GPS Antenna Fail = Ok

0 = Max speed exceeded = speed ok

1 = Tampering sensor = Open

0 = Sleep = No

1 = Battery charging = Yes

0 = Battery fault = No

0 = Power fail = No

0 = Waypoint restricted = Off

c0 = 11000000

0 = Waypoint entering = No

0 = Waypoint leaving = No

0 = GSM jamming = No

00 = Anti-theft status = Disarmed

0 = Accessory missing = No

1 = Moving Status = Moving

1 = Cell ID present = Yes

00 - Speed

40 - Input_mask/Acc_count

0000 = input_mask = no input

0100 = acc_count = 4 accessory

0200000001008293 - Wpt_ID/wpt_group/acc_info

02000000 = Waypoint ID = 2

0100 = Waypoint group = 1

8293 - accelerometer info

82 - accelerometer event

1000 = Side reached

0010 = highest value (/2) = 1.0G

93 - accelerometer value

1001 = Side average (/10) = 0.9G

0011 = Rolling average (/10) = 0.3G

61233000a1d10c25 - 1 wireless packet

61233000 = ID = 3154785

a1 = Device type = MXT101+

d = RSSI = 13

1 = Protocol = 8

0c25 = Data = 0000110000100101

0 = GPRS indicator = No GPRS

000 = Batery Level = <12.5%

11 = Button status = Short pressed 3 times in 5 seconds

00 = Dummy

00100101 = Temperature = 37

18591500a7d10101 - 2 wireless packet

18591500 = ID = 1399064

a7 = Device type = MXT141

d = RSSI = 13

1 = Protocol = 8

0101 = Data = 0000000100000001

0 = Ignition status = off

00 = Button status = no pressed

0 = Door status = closed

000 = Antitheft Mode = normal

01 = High-side output = activated

00 = Low-side output1 = not activated

00 = Low-side output2 = not activated

0 = Under temperature = normal

0 = Over temperature = normal

1 = GPRS indicator = connected

61233000a1e10c25 - 3 wireless packet

61233000 = ID = 3154785

a1 = Device type = MXT101+

e = RSSI = 14

1 = Protocol = 8

0c25 = Data = 0000110000100101

0 = GPRS indicator = No GPRS

000 = Batery Level = <12.5%

11 = Button status = Short pressed 3 times in 5 seconds

00 = Dummy

00100101 = Temperature = 37

01000000f2000000 - 4 wireless packet

01000000 = ID = 1

f2 = Device type = WT110 TAG status

000000 = status = absent

0a012d194a000028 - GPS info/life meter/temperature

0a = SVN = 10

01 = HDOP = 1

2d = SNR = 45

19 = CSQ = 25

004a = Life meter = 74min

00 = Input Voltage = 0

28 = Internal Temperature = 40

7e1a0000 - odometer

00001a7e = 6782m

090a0000 - hourmeter

00000a09 = 2569min

04000000 - posistion sending reason

00000004 - transmission interval moving

342c0000 - voltage info

2c34 = detail info of the voltage = 11.316v

0000 = battery used = 0min

758b0000 - driver ID

00008b75 = 35701

f427500077026309 - cell info

F427 = Country code = 724

5000 = Network code = 5

7702 = Local area code = 277

6309 = Cell ID = 963

cd00 - CRC

04 - End of Frame

3.3.5 Command from server

Server can send this command to MXT device to require current position.

MT: 0x31
DATA FIELD: NONE

3.4 Read configuration

MT: 0x32;
DATA FIELD: 1 byte;

UINT8 mode;
The mode descriptions:

- 0x1 - CRM_DEV_INFO, read device information
- 0x2 - CRM_NET_ATTRIB, read data connection configurations
- 0x3 - CRM_IP_ADDRESS, read IP configurations
- 0x4 - CRM_REPORT_INTERVAL, read position packet reporting configurations
- 0x5 - CRM_GSR, read accelerometer configurations
- 0x6 - CRM_GPS, read GPS configurations
- 0x7 - CRM_SMS, read SMS configurations
- 0x8 - CRM_ZIG_INFO, read 2.4Ghz configurations
- 0x9 - CRM_OTHERS, read other configurations
- 0xC - CRM_PANIC_MODE, read panic mode
- 0xD - CRM_ZIG_INFO_EXT, read extend 2.4Ghz information
- 0xE - CRM_NET_ATTRIB2, read secondary data connection configurations
- 0x80 - CRM_ALL_CFG, read all configurations
- 0xFE - CRM_ALL_CFG, read all configurations for MXT140/141/100N/101N

Reply format is:
MT:0x32
DATA FIELD: variable, please see the following:

For MXT-140/141 and MXT-100N/101N, please see section **3.4.12**
For MXT-142 and MXT-130 devices, please see section **3.25**

3.4.1 Reply of CRM_DEV_INFO

UINT8	mode:	//0x1
UINT32	deviceID:	
UINT8	sw_version[4]:	//if the version is 1.23 then the byte in sw_version is “0123”

Command Example:

01a83223a224001021315904

Reply:

01a83223a22400102123a2240030313130528804

1021 – mode

23a22400 – deviceID

30313130 – sw_version (ASCII)

3.4.2 Reply of CRM_NET_ATTRIB

UINT8	mode:	//0x2
UINT8	connectionType:	//0 is UDP 1 is TCP
UINT16	keepAliveTimer:	//keep alive timer for UDP
UINT8	apnSize:	
UINT8*	Apn:	//apnSize bytes
UINT8	userSize:	
UINT8*	User:	//userSize bytes
UINT8	pswSize:	
UINT8*	Password:	//pswSize bytes

Command example:

01a83223a2240002526904

Reply:

01a83223a22400020000001567656e65726963612e636c61726f2e636f6d2e627205636c61726f05636c61726f525b04

02 – mode

00 – connectionType

0000 – KeepAliveTimer

15 – apnSize

67656E65726963612E636C61726F2E636F6D2E6272 – apn

05 – userSize

636C61726F – user

05 – pwdSize

636C61726F – password

3.4.3 Reply of CRM_IP_ADDRESS

UINT8	mode:	//0x3
UINT8	primaryIPaddrSize:	
UINT8*	primaryIPaddr:	//primaryIPaddrSize bytes
UINT8	secondaryIPaddrSize:	
UINT8*	secondaryIPaddr:	//secondaryIPaddrSize bytes
UINT16	primaryIPport:	
UINT16	secondaryIPport:	

Command example:

01a83223a224001023737904

Reply:

01a83223a22400031567776d746b2e6d6178747261636b2e636f6d2e62721567776d746b2e6d6178747261636b2e636f6d2e62723a163a16673104

03 – mode

15 – primaryIPaddrSize

67776d746b2e6d6178747261636b2e636f6d2e6272 – primaryIPaddr

15 – secondaryIPaddrSize

67776d746b2e6d6178747261636b2e636f6d2e6272 – secondaryIpaddr

3a16 – primaryPort

3a16 – secondaryPort

3.4.4 Reply of CRM_REPORT_INTERVAL

UINT8	mode:	//0x4
UINT16	timerOfIgnitionOff:	//timer for reporting position packets when device is stopped.
UINT16	timerOfMovement:	//timer for reporting position packets when device is moving
UINT16	timerOfPanic:	//timer for reporting position packets when device is in panic status
UINT8	reSendAttempts:	//attempts to send again when sending packet failed or ACK from server was not received
UINT16	reSendTimeout:	//timer of waiting for ACK from server after a packet is sent
UINT8	infoGroup:	
UINT8	keepworkingtimer:	//keep working timer after ignition off, if configure to 0

		means not care this parameter. The timer is in hours.
UINT16	timeoutMoving:	//After G-sensor changes from stopped to moving the unit will send according to moving interval. When G-sensor changes from moving to stopped, the unit will continue to send according to moving interval during “timeoutMoving” minutes. After this period, the unit will transmit as stopped interval.

Command example:
01a83223a224001024940904

Reply:
01a83223a22400102419001900190003 0a0064ff0000893a04
1024 – mode
1900 – timerOfIgnitionOff
1900 – timerOfMovement
1900 – timerOfPanic
03 – reSendAttempts
0a00 – reSendTimeout
64 – infoGroup
ff – keepworkingtimer
0000 – timeoutMoving

3.4.5 Reply of CRM_GSR

UINT8	mode:	//0x5
UINT8	sendImmediately:	//1 indicates that when device detects a change in stopped or moving status, a position packet is sent immediately.
UINT16	debMoving:	//debounce timer for checking if device is moving
UINT16	debStopped:	//debounce timer for checking if device is stopped
UINT16	detectMoving:	//timer to detect the device new state when it is moving
UINT8	rsMode:	//Which is MXT15X position inside vehicle (rolling axis/side axis): X/Y, X/Z, Y/X, Y/Z, Z/X, Z/Y
UINT8	rollingThreshold:	//Rolling axis threshold in number of G’s/10 (0~1.6G=0~16 units range)
UINT8	sideThreshold:	//Side axis threshold in number of G’s/10 (0~1.6G=0~16 units range)
UINT8	shockThreshold:	//Shock threshold in number of G’s/2 (2~16 units = 1~8G range, applies to all axis)

Command example:
01a83223a2240005b51904

Reply:
01a83223a224000500050005000a00000a0a0aa92104
05 – mode
00 – sendImmediately
0500 – debMoving
0500 – debStopped
0a00 – detectMoving
00 – rsMode
0a – rollingThreshold
0a – sideThreshold
0a – shockThreshold

3.4.6 Reply of CRM_GPS

UINT8	mode:	//0x6
UINT16	keepWorkingTimer:	//timer for keeping GPS working after position packet is sent
UINT16	unFixTimeout:	//max timer for waiting for GPS fix
UINT16	unfixColdTimeout:	//max timer for waiting for GPS fix when cold started
UINT16	openBfTransMove:	//timer to turn on GPS before transmission when moving
UINT16	openBfTransStop:	//timer to turn on GPS before transmission when stopped
UINT8	accFilter:	//the accelerate of GPS speed filter, from 4 to 255, in km/h
UINT8	agpsServerIPaddrSize:	
UINT8*	agpsServerIPaddr:	//agpsServerIPaddrSize bytes
UINT16	agpsServerIPport:	

Command example:
01a83223a2240006d62904

Reply:
01a83223a224000600008403b010241e005a002300000010248804
06 – mode
0000 – keepWorkingTimer
8403 – unFixTimeout
B01024 – unfixColdTimeout

1e00 – openBfTransMove

5a00 – openBfTransStop

23 – accFilter

00 – agpsServerIPaddrSize

0000 – agpsServerIPport

3.4.7 Reply of CRM_SMS

UINT8	mode:	//0x7
UINT8	sendMode:	//mode of sending position packets through SMS
UINT8	aliasName[32]:	//alias of the device, this alias will be included in SMS header and sent to server.
UINT8	destination[16]:	//destination phone number for device to send short message to
UINT8	allowNumberMode:	//set allow number to 0(any number), or 1(only destination number)
UINT8	smsMaxSendCount:	//max sending count when no gprs
UINT16	smsInterval:	//sending interval when no gprs
UINT8	smsNumber1[16]:	//three SMS destinations (numbers) to send SMS
UINT8	smsNumber2[16]:	//with position packet only when entering in
UINT8	smsNumber3[16]:	//panic/alarmed status.

Command example:

01a83223a2240007f73904

Reply:

01a83223a2240007024d6178747261636b00000000000000000000000000000000000000000000000033313333313132393030000000000000643c00333133333131323932340000000000003331333331313239323300000000000033313333331313239323200000000000851404

07 – mode

02 – sendMode

4d6178747261636b00000000000000000000000000000000000000000000000000 - aliasName

33313333311313239303000000000000 – destination

00 – allowNumberMode

64 – smsMaxSendCount

3c00 – smsInterval

33313333311313239323400000000000 – smsNumber1

33313333311313239323300000000000 – smsNumber2

33313333311313239323200000000000 – smsNumber3

3.4.8 Reply of CRM_ZIG_INFO

UINT8	mode:	//0x8
UINT8	masterSleepEnable:	//1 is enabled, 0 is disabled
UINT8	keepAliveInterval:	//timer for communication between master and accessory, in minutes
UINT8	keepAliveDuration:	//timer for master to wait for reports from accessory
UINT8	accRetry:	//attempts for accessory to send report.
UINT16	accRxTimeout:	//max timer of accessory for waiting master's ACK
UINT8	accEncryptKey[16]:	//encrypt key for every packet transmitted between 2.4Ghz devices

Command example:

01a83223a224000818c804

Reply

01a83223a22400080005021021f4102110210203102405060708090a0b0c0d0e0f00d2fb04

08 – mode

00 – masterSleepEnable

05 – keepAliveInterval

02 – keepAliveDuration

1021 – accRetry

f41021 – accRxTimeout

10210203102405060708090a0b0c0d0e0f00 – accEncryptKey

3.4.9 Reply of CRM_OTHERS

UINT8	mode:	//0x9
UINT8	ledEnable:	//1 is enabled, 0 is disabled
UINT8	charingOnly:	//1 is enabled, 0 is disabled
UINT8	input1Enable:	//1 is enabled, 0 is disabled
UINT8	input2Enable:	//1 is enabled, 0 is disabled
UINT8	input3Enable:	//1 is enabled, 0 is disabled
UINT8	lowPowerAlert:	//1 is enabled, 0 is disabled
UINT8	timezone:	//0~25, 0 is -12, 12 is 0, 25 is +13
UINT8	maxSpeedLimit:	//max speed limit
UINT32	odometer:	//initial odometer in meters
UINT8	cellInfo:	//1 is "add cell ID to all position packets", 0 is "add only when there is no actual GPS position"

UINT8	dtmfPassword[8]:	//DTMF password
UINT8	input4Enable:	//1 is enabled, 0 is disabled
UINT32	hourmeter:	//minutes with ignition on
UINT8	bSmartOutput2:	//1 activate smart output2, 0 deactivate
UINT8	antiTheftEnable:	//1 activate, 0 deactivate
UINT8	doorDetect:	//1 enabled, 0 disabled
UINT8	localParkingMode:	//1 enabled, 0 disabled
UINT8	bDisableZigbee:	//1 enabled, 0 disabled zigbee when ignition off
UINT8	bOdometerCalc:	//1 enabled, 0 disabled calculate odometer when ignition off
UINT8	bMicrophoneEnable:	//1 enabled, 0 disabled microphone
UINT8	bMoveTriggerAlarm:	//1 trigger, 0 not trigger alarm mode when detect moving
UINT8	eventSentSel[16]:	//select the event can be created.
UINT8	ignDebTimer:	//debounce timer for ingition
UINT8	panicDebTimer:	//debounce timer for panic
UINT8	ipt1DebTimer:	//debounce timer for input1
UINT8	ipt2DebTimer:	//debounce timer for input2
UINT8	ipt3DebTimer:	//debounce timer for input3
UINT8	ipt4DebTimer:	//debounce timer for input4
UINT8	maxSpeedDebTimer:	//debounce timer for max speed detect
UINT8	wpDebTimer:	//debounce timer for waypoint
UINT8	exPowerDebTimer:	//debounce timer for external power detect
UINT8	opt1Mask:	//mask flag for output1
UINT8	opt2Mask:	//mask flag for output2
UINT8	opt3Mask:	//mask flag for output3
UINT16	oldPosTransCount:	//the max count of the old position in log can be transmitted.
UINT8	backdoor:	//0 is reset, 1 is power off
UINT8	ignVolThreshold:	//the voltage threshold of measuring ignition, 0.5v per unit
UINT8	opt1Invert:	//enable or disable output1 invert, 0 is disable, 1 is enable
UINT8	chargingIgnOff:	//enable or disable charging when ignition off
UINT8	speakerEnable:	//enable or disable speaker.
UINT8	odo_rpmEnable:	//0:disable both, 1 enable odometer input only, 2 enable RPM input only, 3 enable both
UINT16	odoPulses:	//pulses per km for odometer measuring
UINT16	rpmPulses:	//pulses per revolution for RPM measuring
UINT16	rpmThreshold:	//set the threshold for RPM
UINT8	buzzerIncomingCall:	//enable or disable buzzer when incoming call
UINT8	cfgAlias[32]:	//the alias name of current configuration

```
UINT8    speakerVol:                //the volume of the speaker(1~10)
UINT8    makeCallTime:              //the timer of outgoing call, in minutes
UINT8    ringMode:                  //incoming call warning
UINT8    incomingNumber[10][20]:    //the incoming call list, 10 numbers
UINT8    outgoingNumber[2][20]:     //the outgoing call list, 2 numbers
UINT8    ignCode:                   //ignition code
UINT8    bBtnDoorAsInput:           //1 is as input, 0 is not
UINT8    gpsFilter:                 //1 means enable filter
UINT8    agpsTimer:                 //start agps interval
UINT8    zigLinkTimer:              //link to another device's interval
UINT8    neutralMaxSpeed:           //max speed limit on neutral
UINT16   neutralRpm:                //rpm limit on neutral
UINT16   neutralExcessiveRpm:       //excessive rpm on neutral
UINT8    gpsFailureDebounveTimer:   //in minutes
UINT8    gpsFailureDebounveTimer:   //in minutes
UINT8    linkFailOpt:               //which output on link fail
UINT8    exceedSpeedOpt:            //which output when exceed speed
UINT8    JammingOpt:                //which output on GSM jamming
UINT8    JammingAlert:              //set alert through 2.4Ghz when Jamming
UINT8    sirenActDuration:          //in seconds
UINT8    sirenDeactDuration:        //in seconds
UINT8    sirenCircle:               //cycles of siren
UINT8    sirenOfOpt:                //1 is output2; 2 is output 3
UINT8    distanceThreshold:         //in hundred meters
UINT8    directionThreshold:        //from 10 to 180 degrees
UINT8    ipPriority:                //0-None, 1-IP1, 2-IP2
UINT8    linkFailTimes:             //1Byte
UINT8    packetEncryptKey[16]:      //encrypt key for all positions packets
UINT8    alertTimer:                //1byte, in seconds
UINT8    rsRollingDeb:              //1Byte, in seconds
UINT8    rsSideDeb:                 //1Byte, in seconds
UINT8    rsShockDeb:                //1Byte, in seconds
UINT8    bGandOSpeed:               //1Byte
UINT8    parkingOpt:                //set anti-theft parking output
UINT8    rollingOpt:                //1Byte, from 0~3
UINT8    sideOpt:                   //1Byte, from 0~3
UINT8    shockOpt:                  //1Byte, from 0~3
UINT8    bGrowingSending:           //1Byte, 1-growing, 0- descending
UINT8    bModifyApn:                //1Byte, 1-enable, 0 disable
UINT16   zigTagInterval:            //2bytes, in seconds
UINT8    panicVia:                  //1-Input1, 2-Wireless Button, 3-Both
```

UINT8	tagFailOpt:	//0: None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	bApOptAllow:	//1byte, 0-disable, 1-enable
UINT8	debSpeedNeutral:	//1Byte
UINT8	debMaxExcessiveRpm:	//1Byte
UINT8	debMaxExcessiveRpmNeutral:	//1Byte
UINT8	parkingOptAct:	//1Byte
UINT8	parkingOptDeact:	//1Byte
UINT8	parkingOptCircle:	//1Byte
UINT8	parkingOptInterval:	//1Byte
UINT8	bGpsForAlert:	//1Byte
UINT8	rpmFactor:	//0-multiplied by 1, 1-multiplied by 10, 2-multiplied by 100, 3-multiplied by 1000
UINT8	GsrDebFactor:	//0: multiplied by 1, 1-multiplied by 10, 2-multiplied by 100
UINT8	rpmUpTrigger:	//1byte, from 0 to 1
UINT8	rpmEvt1Opt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	rpmEvt2Opt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	rpmEvt3Opt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	improperMovingOpt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	anyWt110:	//0-false, 1-true
UINT8	ignLongTimer:	//1byte, in minutes

3.4.10 Reply of CRM_PANIC_MODE

UINT8	mode:	//0xC
UINT8	panicMode:	//0 no trigger, 1 key release trigger, 2 long press key trigger

For MXT-142 and MXT-130 devices, please see section **3.25**

Command example:

01a83223a224000c9c8804

Reply:

01a83223a224000c00808c04

0c – mode

00 – panicMode

3.4.11 Reply of CRM_ZIG_INFO_EXT

UINT8	mode:	//0xD
UINT8	apPowerLevel:	//1-16, the zigbee power level
UINT8	apRadioChan:	//1-4, Zigbee radio channel
UINT8	edRelinkInterval:	//the sync device's link retry interval in minutes
UINT8	edCmdInterval:	//the sync device's command sending interval in hundred million seconds
UINT8	apRevAllBc:	//1 is receiving all broadcast message; 0 is only receiving the message in the list
UINT16	edTimeRetryMax:	//the max retry duration time of the device in minutes
UINT32	linkToken:	//link token (RFU)
UINT8	updByZigbee:	//0 is disable and 1 is enable
UINT8	timeWithoutGsm:	//On the absence GSM/GPRS signal during "timeWithoutGsm" minutes, will activate "Zigbee Alert".
UINT8	zigAlarmTime:	//Zigbee alert message send interval
UINT8	zigRssiFilter:	//the filter of RSSI for the accessories.
UINT8	zigAlarmInfo[18]:	//the message will be broadcast

For MXT-142 and MXT-130 devices, please see section **3.25**

Command example:
01a83223a224000dbd9804

Reply:
01a83223a224000d10300005030214000000000000000000000000000000000000000000000000000000000bab204
0d – mode
1030 – apPowerLevel
00 – apRadioChan
05 – edRelinkInterval
03 – edCmdInterval
02 – apRevAllBc
1400 – edTimeRetryMax
00000000 – linkToken
00 – updByZigbee
00 – timeWithoutGsm
00 – zigAlarmTime
00 – zigRssiFilter
000000000000000000000000000000000000 - zigAlarmInfo

3.4.12 Reply of CRM_ALL_CFG

UINT8	mode:	//0x80
UINT32	deviceID:	
UINT8	sw_version[4]:	//if the version is 1.23 then the byte in sw_version is "0123"
UINT8	connectionType:	//0 is UDP 1 is TCP
UINT16	keepAliveTimer:	//keep alive timer for UDP
UINT8	apnSize:	
UINT8*	Apn:	//apnSIze bytes
UINT8	userSize:	
UINT8*	User:	//userSIze bytes
UINT8	pswSize:	
UINT8*	Password:	//pswSIze bytes
UINT8	primaryIPaddrSize:	
UINT8*	primaryIPaddr:	//primaryIPaddrSize bytes
UINT8	secondaryIPaddrSize:	
UINT8*	secondaryIPaddr:	//secondaryIPaddrSize bytes
UINT16	primaryIPport:	
UINT16	secondaryIPport:	
UINT16	timerOfIgnitionOff:	//timer for reporting position packets when device is stopped.
UINT16	timerOfMovement:	//timer for reporting position packets when device is moving
UINT16	timerOfPanic:	//timer for reporting position packets when device is in panic status
UINT8	reSendAttempts:	//attempts to send again when sending packet failed or ACK from server was not received
UINT16	reSendTimeout:	//timer of waiting for ACK from server after a packet is sent
UINT8	infoGroup:	
UINT8	sendImmediately:	//1 indicates that when device detects a change in stopped or moving status, a position packet is sent immediately.
UINT16	debMoving:	//debounce timer for checking if device is moving
UINT16	debStopped:	//debounce timer for checking if device is stopped
UINT16	detectMoving:	//timer to detect the device new state when it is moving
UINT16	keepWorkingTimer:	//timer for keeping GPS working after position packet is sent

UINT16	unFixTimeout:	//max timer for waiting for GPS fix
UINT16	unfixColdTimeout:	//max timer for waiting for GPS fix when cold started
UINT16	openBfTransMove:	//timer to turn on GPS before transmission when moving
UINT16	openBfTransStop:	//timer to turn on GPS before transmission when stopped
UINT8	sendMode:	//mode of sending position packets through SMS
UINT8	aliasName[32]:	//alias of the device, this alias will be included in SMS header and sent to server.
UINT8	destination[16]:	//destination phone number for device to send short message to
UINT8	allowNumberMode:	//set allow number to 0(any number), or 1(only destination number)
UINT8	masterSleepEnable:	//1 is enabled, 0 is disabled
UINT8	keepAliveInterval:	//timer for communication between master and accessory, in minutes
UINT8	keepAliveDuration:	//timer for master to wait for reports from accessory
UINT8	accRetry:	//attempts for accessory to send report.
UINT16	accRxTimeout:	//max timer of accessory for waiting master's ACK
UINT8	accEncryptKey[16]:	//encrypt key for every packet transmitted between 2.4Ghz devices
UINT8	apPowerLevel:	//1-16£ - the zigbee power level
UINT8	apRadioChan:	//1-4, Zigbee radio channel
UINT8	edRelinkInterval:	//the sync device's link retry interval in minutes
UINT8	edCmdInterval:	//the sync device's command sending interval in hundred million seconds
UINT8	apRevAllBc:	//1 is receiving all broadcast message: 0 is only receiving the message in the list
UINT16	edTimeRetryMax:	//the max retry duration time of the device in minutes
UINT32	linkToken:	//link token(RFU)
UINT8	panicMode:	//0 no trigger, 1 key release trigger, 2 long press key trigger
UINT8	ledEnable:	//1 is enabled, 0 is disabled
UINT8	charingOnly:	//1 is enabled, 0 is disabled
UINT8	input1Enable:	//1 is enabled, 0 is disabled
UINT8	input2Enable:	//1 is enabled, 0 is disabled
UINT8	input3Enable:	//1 is enabled, 0 is disabled
UINT8	lowPowerAlert:	//1 is enabled, 0 is disabled
UINT8	timezone:	//0~25, 0 is -12, 12 is 0, 25 is +13
UINT8	maxSpeedLimit:	//max speed limit
UINT32	odometer:	//initial odometer in meters
UINT8	cellInfo:	//1 is "add cell ID to all position packets", 0 is "add only when there is no actual GPS position"

UINT8	dtmfPassword[8]:	//DTMF password
UINT8	input4Enable:	//1 is enabled, 0 is disabled
UINT32	hourmeter:	//minutes with ignition on
UINT8	bSmartOutput2:	//1 activate smart output2, 0 deactivate
UINT8	antiTheftEnable:	//1 activate, 0 deactivate
UINT8	doorDetect:	//1 enabled, 0 disabled
UINT8	localParkingMode:	//1 enabled, 0 disabled
UINT8	bDisableZigbee:	//1 enabled, 0 disabled zigbee when ignition off
UINT8	bOdometerCalc:	//1 enabled, 0 disabled calculate odometer when ignition off
UINT8	bMicrophoneEnable:	// 1 enabled, 0 disabled microphone
UINT8	bMoveTriggerAlarm:	// 1 trigger, 0 not trigger alarm mode when detect moving
UINT8	eventSentSel[16]:	//select the event can be created.
UINT8	ignDebTimer:	//debounce timer for ingition
UINT8	panicDebTimer:	//debounce timer for panic
UINT8	ipt1DebTimer:	//debounce timer for input1
UINT8	ipt2DebTimer:	//debounce timer for input2
UINT8	ipt3DebTimer:	//debounce timer for input3
UINT8	ipt4DebTimer:	//debounce timer for input4
UINT8	maxSpeedDebTimer:	//debounce timer for max speed detect
UINT8	wpDebTimer:	//debounce timer for waypoint
UINT8	exPowerDebTimer:	//debounce timer for external power detect
UINT8	opt1Mask:	//mask flag for output1
UINT8	opt2Mask:	//mask flag for output2
UINT8	opt3Mask:	//mask flag for output3
UINT16	oldPosTransCount:	//the max count of the old position in log can be transmitted.
UINT8	backdoor:	//0 is reset, 1 is power off
UINT8	ignVolThreshold:	//the voltage threshold of measuring ignition, 0.5v per unit
UINT8	opt1Invert:	//enable or disable output1 invert, 0 is disable, 1 is enable
UINT8	chargingIgnOff:	//enable or disable charging when ignition off
UINT8	accFilter:	//the accelerate of GPS speed filter, from 4 to 255, in km/h
UINT8	speakerEnable:	//enable or disable speaker.
UINT8	odo_rpmEnable:	//0:disable both, 1 enable odometer input only, 2 enable RPM input only, 3 enable both
UINT16	odoPulses:	//pulses per km for odometer measuring
UINT16	rpmPulses:	//pulses per revolution for RPM measuring
UINT16	rpmThreshold:	//set the threshold for RPM

UINT8	rsMode:	//Which is MXT15X position inside vehicle (rolling axis/side axis): X/Y, X/Z, Y/X, Y/Z, Z/X, Z/Y
UINT8	rollingThreshold:	//Rolling axis threshold in number of G's/10 (0~1.6G=0~16 units range)
UINT8	sideThreshold:	//Side axis threshold in number of G's/10 (0~1.6G=0~16 units range)
UINT8	shockThreshold:	//Shock threshold in number of G's/2 (2~16 units = 1~8G range, applies to all axis)
UINT8	smsMaxSendCount:	//max sending count when no gprs
UINT16	smsInterval:	//sending interval when no gprs
UINT8	smsNumber1[16]:	//three SMS destinations (numbers) to send SMS
UINT8	smsNumber2[16]:	/ with position packet only when entering in
UINT8	smsNumber3[16]:	//panic/alarmed status.
UINT8	buzzerIncomingCall:	//enable or disable buzzer when incoming call
UINT8	cfgAlias[32]:	//the alias name of current configuration
UINT8	keepworkingtimer:	//keep working timer after ignition off, if configure to 0 means not care this parameter. The timer is in hours.
UINT8	speakerVol:	//the volume of the speaker(1~10)
UINT8	makeCallTime:	//the timer of outgoing call, in minutes
UINT8	ringMode:	//incoming call warning
UINT8	incomingNumber[10][20]:	//the incoming call list, 10 numbers
UINT8	outgoingNumber[2][20]:	//the outgoing call list, 2 numbers
UINT8	agpsServerIPaddrSize:	
UINT8*	agpsServerIPaddr:	//agpsServerIPaddrSize bytes
UINT16	agpsServerIPport:	
UINT8	updByZigbee:	//0 is disable and 1 is enable
UINT16	timeoutMoving:	//After G-sensor changes from stopped to moving the unit will send according to moving interval. When G-sensor changes from moving to stopped, the unit will continue to send according to moving interval during "timeoutMoving" minutes. After this period, the unit will transmit as stopped interval.
UINT8	timeWithoutGsm:	//On the absence GSM/GPRS signal during "timeWithoutGsm" minutes, will activate "Zigbee Alert".
UINT8	zigAlarmTime:	//Zigbee alert message send interval
UINT8	zigRssiFilter:	//the filter of RSSI for the accessories.
UINT8	zigAlarmInfo[18]:	//the message will be broadcast
UINT8	ignCode:	//ignition code
UINT8	bBtnDoorAsInput:	//1 is as input, 0 is not
UINT8	gpsFilter:	//1 means enable filter
UINT8	agpsTimer:	//start agps interval

UINT8	zigLinkTimer:	//link to another device's interval
UINT8	neutralMaxSpeed:	//max speed limit on neutral
UINT16	neutralRpm:	//rpm limit on neutral
UINT16	neutralExcessiveRpm:	// excessive rpm on neutral
UINT8	gpsFailureDebounveTimer:	//in minutes
UINT8	linkFailOpt:	//which output on link fail
UINT8	exceedSpeedOpt:	//which output when exceed speed
UINT8	JammingOpt:	//which output on GSM jamming
UINT8	JammingAlert:	//set alert through 2.4Ghz when Jamming
UINT8	sirenActDuration:	//in seconds
UINT8	sirenDeactDuration:	//in seconds
UINT8	sirenCircle:	//cycles of siren
UINT8	sirenOfOpt:	//1 is output2; 2 is output 3
UINT8	distanceThreshold:	//in hundred meters
UINT8	directionThreshold:	//from 10 to 180 degrees
UINT8	ipPriority:	//0-None, 1-IP1, 2-IP2
UINT8	linkFailTimes:	//1Byte
UINT8	packetEncryptKey[16]:	//encrypt key for all positions packets
UINT8	apnSize:	//0~99
UINT8*	Apn:	//apnSIze bytes
UINT8	userSize:	//0~31
UINT8*	User:	//userSIze bytes
UINT8	pswSize:	//0~31
UINT8*	Password:	//pswSIze bytes
UINT8	alertTimer:	//1byte, in seconds
UINT8	rsRollingDeb:	//1Byte, in seconds
UINT8	rsSideDeb:	//1Byte, in seconds
UINT8	rsShockDeb:	//1Byte, in seconds
UINT8	bGandOSpeed:	//1Byte
UINT8	parkingOpt:	//set anti-theft parking output
UINT8	rollingOpt:	//1Byte, from 0~3
UINT8	sideOpt:	//1Byte, from 0~3
UINT8	shockOpt:	//1Byte, from 0~3
UINT8	bGrowingSending:	//1Byte, 1-growing, 0- descending
UINT8	bModifyApn:	//1Byte, 1-enable, 0 disable
UINT16	zigTagInterval:	//2bytes, in seconds
UINT8	panicVia:	//1-Input1, 2-Wireless Button, 3-Both
UINT8	tagFailOpt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	bApOptAllow:	//1byte, 0-disable, 1-enable
UINT8	debSpeedNeutral:	//1Byte
UINT8	debMaxExcessiveRpm:	//1Byte

UINT8	debMaxExcessiveRpmNeutral:	//1Byte
UINT8	parkingOptAct:	//1Byte
UINT8	parkingOptDeact:	//1Byte
UINT8	parkingOptCircle:	//1Byte
UINT8	parkingOptInterval:	//1Byte
UINT8	bGpsForAlert:	//1Byte
UINT8	rpmFactor:	//0-multiplied by 1, 1-multiplied by 10, 2-multiplied by 100, 3-multiplied by 1000
UINT8	GsrDebFactor:	//0-multiplied by 1, 1-multiplied by 10, 2-multiplied by 100
UINT8	rpmUpTrigger:	//1byte, from 0 to 1
UINT8	rpmEvt1Opt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	rpmEvt2Opt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	rpmEvt3Opt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	improperMovingOpt:	//0-None, 1-Opt1, 2-Opt2, 3-Opt3
UINT8	anyWt110:	//0-false, 1-true
UINT8	ignLongTimer:	//1byte, in minutes

This sub-command was replaced by 0xFE
For all devices, use this command: **01A03200000000FE126F04**

For MXT-142 and MXT-130 devices, please see section **3.25**

Command with NO PARAMETER

Reply of 0xFE:

UINT32	deviceID	//4 Bytes
UINT8	firstVersion	//1 Byte
UINT8	secondVersion	//1 Byte
UINT32	buildtime	//4 Bytes
UINT8	productnamesize	//1 Byte
UINT8*	productname	//productnamesize Bytes
UINT8	appfirstVersion	//1 Byte
UINT8	appsecondVersion	//1 Byte
UINT32	appbuildtime	//4 Bytes
UINT8	appnamesize	//1 Byte
UINT8*	appname	//appnamesize Bytes
UINT8	pinstatus	//1 Byte
UINT8	zigbeeversionsize	//1 Byte

UINT8*	zigbeeversion	//zigbeeversionsize Bytes
UINT32	zigbeebuildtime	//4 Bytes
	IMSI	//32 Bytes
	IMEI	//20 Bytes
UINT32	MTC_CFG1	//4 Bytes
UINT32	MTC_CFG2	//4 Bytes
UINT8	reSendCount	//1 Byte
UINT8	accRetry	//1 Byte
UINT8	kaInterval	//1 Byte
UINT8	kaDuration	//1 Byte
UINT8	apPowerLevel	//1 Byte
UINT8	apRadioChan	//1 Byte
UINT8	edRelinkInterval	//1 Byte
UINT8	edCmdInterval	//1 Byte
UINT8	apRevBcOpt	//1 Byte
UINT8	maxSpeed	//1 Byte
UINT8	infoGroup	//1 Byte
UINT8	timeZone	//1 Byte
UINT8	ignDebTimer	//1 Byte
UINT8	panicDebTimer	//1 Byte
UINT8	ipt1DebTimer	//1 Byte
UINT8	ipt2DebTimer	//1 Byte
UINT8	ipt3DebTimer	//1 Byte
UINT8	ipt4DebTimer	//1 Byte
UINT8	maxSpeedDebTimer	//1 Byte
UINT8	wpDebTimer	//1 Byte
UINT8	exPowerDebTimer	//1 Byte
UINT8	ignVolLimit	//1 Byte
UINT8	accFilter	//1 Byte
UINT8	rollingThreshold	//1 Byte
UINT8	sideThreshold	//1 Byte
UINT8	shockThreshold	//1 Byte
UINT8	smsMaxSendCount	//1 Byte
UINT8	keepWorkingBfSleep	//1 Byte
UINT8	makeCallTime	//1 Byte
UINT8	timeWithoutGsm	//1 Byte
UINT8	zigAlarmTime	//1 Byte
UINT8	zigRssiFilter	//1 Byte
UINT8	ignCode	//1 Byte
UINT16	keepAliveTimer	//2 Bytes
UINT16	primaryPort	//2 Bytes

UINT16	secondaryPort	//2 Bytes
UINT16	ignitionOff	//2 Bytes
UINT16	movement	//2 Bytes
UINT16	panic	//2 Bytes
UINT16	reSendTimer	//2 Bytes
UINT16	debMoving	//2 Bytes
UINT16	debStopped	//2 Bytes
UINT16	detectingInterval	//2 Bytes
UINT16	keepWorkingTimer	//2 Bytes
UINT16	unfixTimeout	//2 Bytes
UINT16	unfixColdTimeout	//2 Bytes
UINT16	transTimerMove	//2 Bytes
UINT16	transTimerStop	//2 Bytes
UINT16	accRxTimeout	//2 Bytes
UINT16	edTimeRetryMax	//2 Bytes
UINT16	oldPosTransCount	//2 Bytes
UINT16	odoPulses	//2 Bytes
UINT16	rpmPulses	//2 Bytes
UINT16	rpmThreshold	//2 Bytes
UINT16	smsInterval	//2 Bytes
UINT16	agpsPort	//2 Bytes
UINT16	timeoutMoving	//2 Bytes
UINT32	linkToken	//4 Bytes
UINT32	odometer	//4 Bytes
UINT32	hourmeter	//4 Bytes
UINT128	accEncryptKey	//16 Bytes
UINT128	eventSentSel	//16 Bytes
UINT8	passwordlen	//1 Byte
UINT8*	password	//passwordlen Bytes
UINT8	pinlen	//1 Byte
UINT8*	pin	//pinlen Bytes
UINT8	puklen	//1 Byte
UINT8*	puk	//puklen Bytes
UINT8	smspasswordlen	//1 Byte
UINT8*	smspassword	//smspasswordlen Bytes
UINT8	apnlen	//1 Byte
UINT8*	apn	//apnlen Bytes
UINT8	userlen	//1 Byte
UINT8*	user	//userlen Bytes
UINT8	passwordlen	//1 Byte
UINT8*	password	//passwordlen Bytes

UINT8	primaryIPAddrlen	//1 Byte
UINT8*	primaryIPAddr	//primaryIPAddrlen Bytes
UINT8	secondaryIPAddrlen	//1 Byte
UINT8*	secondaryIPAddr	//secondaryIPAddrlen Bytes
UINT8	agpsIPAddrlen	//1 Byte
UINT8*	agpsIPAddr	//agpsIPAddrlen Bytes
UINT8	aliasNamelen	//1 Byte
UINT8*	aliasName	//aliasNamelen Bytes
UINT8	destinationlen	//1 Byte
UINT8*	destination	//destinationlen Bytes
UINT8	cfgAliaslen	//1 Byte
UINT8*	cfgAlias	//cfgAliaslen Bytes
UINT8	zigAlarmInfolen	//1 Byte
UINT8*	zigAlarmInfo	//zigAlarmInfolen Bytes
UINT8	smsPanicNum1len	//1 Byte
UINT8*	smsPanicNum1	//smsPanicNum1len Bytes
UINT8	smsPanicNum2len	//1 Byte
UINT8*	smsPanicNum2	//smsPanicNum2len Bytes
UINT8	smsPanicNum3len	//1 Byte
UINT8*	smsPanicNum3	//smsPanicNum3len Bytes
UINT8	allowNumber1len	//1 Byte
UINT8*	allowNumber1	//allowNumber1len Bytes
UINT8	allowNumber2len	//1 Byte
UINT8*	allowNumber2	//allowNumber2len Bytes
UINT8	allowNumber3len	//1 Byte
UINT8*	allowNumber3	//allowNumber3len Bytes
UINT8	allowNumber4len	//1 Byte
UINT8*	allowNumber4	//allowNumber4len Bytes
UINT8	allowNumber5len	//1 Byte
UINT8*	allowNumber5	//allowNumber5len Bytes
UINT8	allowNumber6len	//1 Byte
UINT8*	allowNumber6	//allowNumber6len Bytes
UINT8	allowNumber7len	//1 Byte
UINT8*	allowNumber7	//allowNumber7len Bytes
UINT8	allowNumber8len	//1 Byte
UINT8*	allowNumber8	//allowNumber8len Bytes
UINT8	allowNumber9len	//1 Byte
UINT8*	allowNumber9	//allowNumber9len Bytes
UINT8	allowNumber10len	//1 Byte
UINT8*	allowNumber10	//allowNumber10len Bytes
UINT8	makeNumber1len	//1 Byte

UINT8*	makeNumber1	//makeNumber1len Bytes
UINT8	makeNumber2len	//1 Byte
UINT8*	makeNumber2	//makeNumber2len Bytes
UINT8	agpsTimer	//1 Byte
UINT8	zigLinkTimer	//1 Byte
UINT8	neutralMaxSpeed	//1 Byte
UINT16	neutralRpm	//2 Bytes
UINT16	neutralExcessiveRpm	//2 Bytes
UINT8	gpsFailureTimer	//1 Byte
UINT8	sirenActDuration	//1 Byte
UINT8	sirenDeactDuration	//1 Byte
UINT8	sirenCircle	//1 Byte
UINT32	MTC_CFG3	//4 Bytes
UINT8	distanceThreshold	//1 Byte
UINT8	directionThreshold	//1 Byte
UINT8	linkFailTimes	//1 Byte
UINT8	packetEncryptKey[16]	//1 Byte or 16 Bytes
UINT8	apn2len	//1 Byte
UINT8*	apn2	//apn2len Bytes
UINT8	user2len	//1 Byte
UINT8*	user2	//user2len Bytes
UINT8	password2len	//1 Byte
UINT8*	password2	//password2len Bytes
UINT8	alertTimer	//1 Byte
UINT8	rsRollingDeb	//1 Byte
UINT8	rsSideDeb	//1 Byte
UINT8	rsShockDeb	//1 Byte
UINT16	zigTagInterval	//2 Bytes
UINT8	debSpeedNeutral	//1 Byte
UINT8	debMaxExcessiveRpm	//1 Byte
UINT8	debMaxExcessiveRpmNeutral	//1 Byte
UINT8	parkingOptAct	//1 Byte
UINT8	parkingOptDeact	//1 Byte
UINT8	parkingOptCircle	//1 Byte
UINT8	parkingOptInterval	//1 Byte
UINT8	smsSpeedAlertNumlen	//1 Byte
UINT8*	smsSpeedAlertNum	//smsSpeedAlertNumlen Bytes
UINT8	ignLongTimer	//1 Byte
UINT32	MTC_CFG4	//4 Bytes
UINT8	gsrThresholdLevel	//1 Byte
UINT8	rs232WorkingTimer	//1 Byte

```
UINT8           rs232RecvTimeout        //1 Byte
UINT8           rainMaxSpeed            //1 Byte
UINT8           accelerationLimit       //1 Byte
UINT8           decelerationLimit       //1 Byte
UINT32          MTC_CFG5                //4 Bytes
UINT8           maxSpeedGenericEvent    //1 Byte
UINT8           engineSealDeb           //1 Byte
UINT8           engineSealOptActTimer   //1 Byte
UINT8           engineSealOptDeactTimer     //1 Byte

typedef union
{
    u32 data
    struct
    {
        u32     size:                   //11
        u32     mode:                   //1
        u32     cellInPos:              //1
        u32     sendImmediately:        //1
        u32     ledEnable:              //1
        u32     chargingOnly:           //1
        u32     smsMode:                //2
        u32     apSleepEnable:          //1
        u32     input1Enable:           //1
        u32     input2Enable:           //1
        u32     input3Enable:           //1
        u32     input4Enable:           //1
        u32     lowerPowerAlert:        //1
        u32     panicMode:              //2
        u32     smsAllowNumber:         //1
        u32     bSmartOutput2:          //1
        u32     antiTheftEnable:        //1
        u32     doorDetect:             //1
        u32     btnParkingMode:         //1
        u32     bEnableZigbee:          //1
    }cfg;
}MTC_CFG1

typedef union
{
    u32 data
```

```
    struct
    {
        u32    bOdometerCalc:          //1
        u32    bMicrophoneEnable:      //1
        u32    bMoveTriggerAlarm:      //1
        u32    opt1Mask:               //1
        u32    opt2Mask:               //1
        u32    opt3Mask:               //1
        u32    output1Invert:          //1
        u32    chargingIgnOff:         //1
        u32    odo_rpmEnable:          //2
        u32    speakerEnable:          //1
        u32    buzzerIncomingCall:     //1
        u32    updByZigbee:            //1
        u32    rsMode:                 //3
        u32    speakerVol:             //4
        u32    ringMode:               //2
        u32    backdoorFunc:           //1
        u32    bBtnDoorAsInput:        //1
        u32    gpsFilter:              //1
        u32    jammingAlert:           //1
        u32    jammingOpt:             //2
        u32    exceedSpeedOpt:         //2
        u32    linkFailOpt:            //2
    }cfg;
}MTC_CFG2

typedef union
{
    u32data
    struct
    {
        u32    ipPriority:             //2
        u32    sirenOfOpt:             //2
        u32    parkingOpt:             //2
        u32    rollingOpt:             //2
        u32    sideOpt:                //2
        u32    shockOpt:               //2
        u32    bGandOSpeed:            //1
        u32    bGrowingSending:        //1
        u32    bModifyApn:             //1
```

```
        u32     bGpsForAlert:           //1
        u32     panicVia:               //2
        u32     tagFailOpt:             //2
        u32     bApOptAllow:            //1
        u32     rpmUpTrigger:           //1
        u32     rpmFactor:              //2
        u32     gsrDebFactor:           //2
        u32     rpmEvt1Opt:             //2
        u32     rpmEvt2Opt:             //2
        u32     rpmEvt3Opt:             //2
    }cfg
}MTC_CFG3

typedef union
{
    u32data
    struct
    {
        u32     improperMovingOpt:      //2
        u32     anyWt110:               //1
        u32     simchip:                //1
        u32     exWorkingMode:          //2
        u32     ignVolFactor:           //2
        u32     stopIntervalFactor:     //2
        u32     input1Action:           //1
        u32     rs232Speed:             //3
        u32     rs232WorkingMode:       //2
        u32     rs232Alert:             //1
        u32     rs232WkModeEx:          //1
        u32     antiStatInfo:           //1
        u32     antiSilence:            //1
        u32     alarmAfterTimer:        //1
        u32     bindPort:               //1
        u32     rainInput:              //2
        u32     bAnyRfid:               //1
        u32     rfidLogoutMode:         //1
        u32     accelerationOutput:     //2
        u32     decelerationOutput:     //2
        u32     rfidPassengerOpt:       //1
        u32     ipt1Invert:             //1
    }cfg
```

```
}MTC_CFG4

typedef union
{
    u32data;
    struct
    {
        u32    ipt2Invert:          //1
        u32    ipt3Invert:          //1
        u32    genericEventInput:   //2
        u32    genericEventOutput:  //2
        u32    rainSensorSignal:    //1
        u32    dummy:               //25
    }cfg
}MTC_CFG5
```

Command example:

01a03200000000fe126f04

Reply example:

01a632189a7b00fe189a7b000209eed5963c064d585431343002204eaa883c0a4d58543134305f4150501021
00373234303531383132303932393832000000000000000000000000000000000033353533373120303220373
2320393736203200000e65cf800021024051021031021050210300005030 2c8640c0200b4b4ff00000002ff230
a0a0a64ff050000002a00003a163a161e001e001e000a00b400b400681021ffff8403b010241e005a00f4102114
00c80088103302001e003c000000000000000000000000e6410f00102102031024050607080 90a0b0c0d0e
0f000200000000e4f91fd0f3c7ffffff000000000001567656e65726963612e636c61726f2e636f6d2e627205636c
61726f05636c61726f0f3230302e3137302e3133332e3139340f3230302e3137302e3133332e3139340000000
0000000000000000000000000000001e005009000a0000ff00ff08402300002d03000000000f00000000000
00000f0f0ff00001e40200000000a32503232001021000000000102400ee9f04

189a7b00 – device_id

02 – first_version

09 – second_version

eed5963c – build_time

06 – productNameSize

4d5854313430 – productname

02 – appfirstVersion

20 – appsecondVersion

4eaa883c – appbuildtime

0a – appnamesize

4d58543134305f415050 – appname

1021 – pinstatus

00 – zigbeeversionsize

373234303531383132303932393832000000000000000000000000000000000000 – imsi

333535333731203032203732303937362032000 – imei

e65cf800 – MTC_CFG1

021024051021 – MTC_CFG2

03 – reSendCount

1021 – accRetry

05 – kaInterval

02 – kaDuration

1030 – apPowerLevel

00 – apRadioChan

05 – edRelinkInterval

03 – edCmdInterval

02 – apRevBcOpt

c8 – maxSpeed

64 – infoGroup

0c – timeZone

02 – ignDebTimer

00 – panicDebTimer

b4 – ipt1DebTimer

b4 – ipt2DebTimer

ff – ipt3DebTimer

00 – Ipt4DebTimer

00 – maxSpeedDebTimer

00 – wpDebTimer

02 – exPowerDebTimer

ff – ignVolLimit

23 – accFilter

0a – rollingThreshold

0a – sideThreshold

0a – shockThreshold

64 – smsMaxSendCount

ff – keepWorkingBfSleep

05 – makeCallTime

00 – timeWithoutGsm

00 – zigAlarmTime

00 – zigRssiFilter

2a – ignCode

0000 – keepAliveTimer

3a16 – primaryPort

3a16 – secondaryPort

1e00 – igniteonOff

1e00 – movement

1e00 – panic

0a00 – reSendTimer

b400 – debMoving

b400 – debStopped

681021 – detectingInterval

ffff – keepWorkingTimer

8403 – unfixTimeout

b01024 – unfixColdTimeout

1e00 – transTimerMove

5a00 – transTimerStop

f41021 – accRxTimeout

1400 – edTimeRetryMax

c800 – oldPosTransCount

881033 – odoPulses

0200 – rpmPulses

1e00 – rpmThreshold

3c00 – smsInterval

0000 – agpsPort

0000 – timeoutMoving

00000000 – linkToken

00000000 – odometer

e6410f00 – hourmeter

102102031024050607080 90a0b0c0d0e0f00 – accEncryptKey

0200000000e4f91fd0f3c7ffffff0000 – eventSentSel

00 – passwordlen

00 – pinlen

00 – puklen

00 – smspasswordlen

15 – apnlen

67656e65726963612e636c61726f2e636f6d2e6272 – apn

05 – userlen

636c61726f – user

05 – passwordlen

636c61726f – password

0f – primaryIPAddrlen

3230302e3137302e3133332e313934 – primaryIPAddr

0f – secondaryIPAd drlen

3230302e3137302e3133332e313934 – secondaryIPAddr

00 - agpsIPAddrlen

00 – aliasNamelen

00 – destinationlen

00 – cfgAliaslen

00 – zigAlarmInfolen

00 – smsPanicNum1len

00 – sms PanicNum2len

00 – smsPanicNum3len

00 – allowNumber1len

00 – allowNumber2len

00 – allowNumber3len

00 – allowNumber4len

00 – allowNumber5len

00 – allowNumber6len

00 – allowNumber7len

00 – allowNumber8len

00 – allowNumber9len

00 – allowNumber10len

00 – makeNumber1len

00 – makeNumber2len

1e – agpsTimer

00 – zigLinkTimer

50 – neutralMaxSpeed

0900 – neutralRpm

0a00 – neutralExcessiveRpm

00 – gpsFailureTimer

ff – sirenActDuration

00 – sirenDeactDuration

ff – sirenCircle

08402300 – MTC_CFG3

00 – distanceThreshold

2d – directionThreshold

03 – linkFailTimes

00 – packetEncryptKey

00 – apn2len

00 – user2len

00 – password2len

0f – alertTimer

00 – rsRollingDeb

00 – rsSideDeb

00 – rsShockDeb

0000 – zigTagInterval

00 – debSpeedNeutral

00 – debMaxExcessiveRpm

00 – debMaxExcessiveRpmNeutral

f0 – parkingOptAct

f0 – parkingOptDeact

ff – parkingOptCircle

00 – parkingOptInterval

00 – smsSpeedAlertNumlen

1e – ignLongTimer

40200000 – MTC_CFG4

00 – gsrThresholdLevel

0a - rs232WorkingTimer

32 - rs232RecvTimeout

50 – rainMaxSpeed

32 – accelerationLimit

32 – decelerationLimit

0010210000 – MTC_CFG5

00 – maxSpeedGenericEvent

00 – engineSealDeb

1024 – engineSealOptActTimer

00 – engineSealOptDeactTimer

3.4.13 Reply of CRM_NET_ATTRIB2

UINT8	apnSize:	//0~99
UINT8*	Apn:	//apnSIze bytes
UINT8	userSize:	//0~31
UINT8*	User:	//userSIze bytes
UINT8	pswSize:	//0~31
UINT8*	Password:	//pswSIze bytes

3.5 Set configuration

Frame containing a list of specific binary parameters, as listed on this topic. The reply must be an ACK or NACK.
All configurations description, please see chapter 3.4
MT: 0x33
DATA FIELD:

ID (1 BYTE)	PARAMETER (VARIABLE)

Note: this DATA FIELD can be composed by more than one parameter, as long as the whole command is limited to 1K.

ID	PARAMETER	ID	PARAMETER	ID	PARAMETER	…

Therefore the whole parameter must be sent even when it contains null fields.

Set ID descriptions:

0x3 - (SPC_DTMF_PWD): set DTMF password

UINT8	pwdSize:	//the size of the password
UINT8*	password:	//pwdSize bytes, the DTMF password

Command example:
01a6332f167c00031024313233349c1f04
03 – Command ID
1024 – pwdSize
31323334 – password (ASCII Value)

0x4 - (SPC_CONNECT_MODE): set type of connection

UINT8	type;

Command example:
01a6332f167c00102410215b6f04
1024 – Command ID
1021 – type (TCP)

0x5 - (SPC_APN): set APN user and password

UINT8	apnSize:	//apn size, max 100chars
UINT8	userSize:	//user name size, max 32chars
UINT8	pwdSize:	//password size, max 32chars
UINT8*	apn:	//apnSize bytes, the apn

UINT8* user: //userSize bytes, the user name
UINT8* password: //pwdSize bytes, the password

Command example:
01a6332f167c000515050567656e65726963612e636c61726f2e636f6d2e6272636c61726f636c61726f797504
15 – apnSize (21 chars)
05 – userSize
05 – pwdSize
67656e65726963612e636c61726f2e636f6d2e6272 – apn (ASCII value)
636c61726f – user (ASCII value)
636c61726f – password (ASCII value)

0x6 - (SPC_IP_ADDR): set IP and port
UINT8 ipIndex: //0 is primary ip, 1 is secondary ip, 2 is AGPS server ip
UINT8 addrSize: //address size
UINT16 port: //the port of the IP
UINT8* addr: //addrSize bytes, the IP address

Command example:
01a6332f167c000600103034236777 6d746b2e6d78742e636f6d2e6272dcfe04
00 – ipIndex
1030 – addrSize
3423 – port
67776d746b2e6d78742e636f6d2e6272 – addr (ASCII value)

0x7 - (SPC_KEEP_ALIVE_TIMER): set keep alive timer
UINT16 keepAliveTimer

Command example:
01a6332f167c00070c0045e904
0c00 – keepAliveTimer (12sec)

0x8 - (SPC_RI_STOPPED): set report time with ignition off
UINT16 timerOfIgnitionOff;

Command example:
01a6332f167c0008b0102490de04
b01024 – timerOfIgnitionOff (1200sec)

0x9 - (SPC_RI_MOVING): set report time with movement

UINT16 timerOfMovement;

Command example:
01a6332f167c000958027d103004
5802 – timerOfMovement (600sec)

0xA - (SPC_RI_PANIC): set report time with panic
UINT16 timerOfPanic;

Command example:
01a6332f167c000a2c1021d3bd04
2c1021 – timerOfPanic (300sec)

0xB - (SPC_RI_RESEND) set resend attempts and timeout
UINT8 attempts
UINT16 timeout

Command example:
01a6332f167c000b080d00a9cc04
08 – attempts
0d00 - timeout

0xC - (SPC_GSR_DEB_MOVING): set debounce timer when device from moving to stopped.
UINT16 timer;

Command example:
01a6332f167c000c190032e504
1900 – timer (25sec)

0xD - (SPC_GSR_DEB_STOPPED): set debounce timer when device from stopped to moving
UINT16 timer;

Command example:
01a6332f167c000d0f00d77b04
0f00 – timer (15sec)

0xE - (SPC_GSR_DETECT): set the timer for detecting when moving
UINT16 timer;

Command example:
01a6332f167c000e0a0072dd04

0a00 – timer (10sec)

0xF - (SPC_GSR_REPORT): set send immediately or not
UINT8 sendImmediately;

Command example:
01a6332f167c000f1021a1b304
1021 – sendImmediately (true)

0x10 - (SPC_GPS_KEEP_WORKING): set GPS keep working timer
UINT16 timer;

Command example:
01a6332f167c001030b40012b804
b400 – timer (180sec)

0x11 - (SPC_GPS_UNFIX_TIMEOUT): set GPS timeout for fix
UINT16 timer;

Command example:
01a6332f167c0010310807a5a404
0807 – timer (1800sec)

0x12 - (SPC_GPS_COLDSTART_UNFIX_TIMEOUT): set GPS timeout for fix when cold started
UINT16 timer;

Command example:
01a6332f167c00126009b89e04
6009 – timer (2400sec)

0x13 - (SPC_GPS_OPEN_BEF_TRANS): set GPS on before transmission
UINT16 MovingTimer;
UINT16 StoppedTimer;

Command example:
01a6332f167c0010335a00b4007cbf04
5a00 – MovingTimer (90sec)
B400 – StoppedTimer (180sec)

0x14 - (SPC_SMS_ALIAS): set SMS alias
UINT8 aliasSize: //the size of the alias

UINT8* alias: //aliasSize bytes, the alias

Command example:
01a6332f167c0014084d6178747261636beb5404
08 – aliasSize
4d6178747261636b – alias (ASCII value)

0x15 - (SPC_SMS_DESTINATION) set SMS destination number
UINT8 destSize: //the size of the destination number
UINT8* destination: //destSize bytes, the destination number

Command example:
01a6332f167c00150a3331333331313232393030589204
0a – destSize
33313333313132393030 – destination (ASCII value)

0x16 - (SPC_SMS_SEND_MODE): set send SMS mode
UINT8 smsMode;
UINT8 allowNumberMode;

Command example:
01a6332f167c0016102110216bfb04
1021 – smsMode (Send when No GPS)
1021 – allowNumberMode (Only destination)

0x17 - (SPC_ZIG_AP_SLEEP): set master accessory can sleep or not
UINT8 masterSleepEnable: //01 - True, 00 - False

Command example:
01a7336372150017102124250 4
1021 – masterSleepEnable (true)

0x18 - (SPC_ZIG_KA_INTERVAL): set keep alive interval
UINT8 timer;

Command example:
01a73363721500181923a604
19 – timer (25sec)

0x19 - (SPC_ZIG_KA_DURATION): set keep alive duration
UINT8 timer;

Command example:
01a73363721500190fe5e704
0f – timer (15sec)

0x1A - (SPC_ZIG_ACC_RETRY): set accessory attempts for sending reports
UINT8 accRetry;

Command example:
01a733637215001a05fc1304
05 – accRetry

0x1B - (SPC_ZIG_ACC_RX_TIMEOUT): set accessory RX timeout for waiting for master response
UINT16 timeout: //2 Bytes

Command example:
01a733637215001b00198f9504
0019 – timeout (25sec)

0x1C - (SPC_ZIG_ACC_ENCRYPT_KEY): set the encrypt key of data.
UINT8* accEncryptKey: //16Bytes, the accessory encrypt data key

Command example:
01a733637215001c102102031024050607080 90a0b0c0d0e0f00f6a104
102102031024050607080 90a0b0c0d0e0f00 – accEncryptKey (ASCII value)

0x1D - (SPC_LED_ENABLE): set enable led working or not
UINT8 enable;

Command example:
01a733637215001d1021efca04
1021 – enable (true)

0x1E - (SPC_CHARGING_ONLY): set start or stop charging only mode
UINT8 chargingOnly: //1 is started, 0 is stopped

Command example:
01a733637215001e009d8f04
00 – chargingOnly

0x1F - (SPC_INPUT_ENABLE)
UINT8 inputIndex : the index of the input port (1~4)
UINT8 enable: //1 is enable, 0 is disable

Command example:
01a733637215001f02001f03009c0804
02 – inputIndex (input 2)
00 – enable
03 – inputIndex (input 3)
00 - enable

0x20 - (SPC_OUPUT_ACTIVATE)
UINT8 outputIndex : the index of the output port (1~3)
UINT8 activate: //activate or deactivate

Command example:
01a7336372150020021021805504
02 – outputIndex
1021 – activate

0x21 - (SPC_TIME_ZONE): set the time zone
UINT8 timezone: //from -12 to +13

Command example:
01a73363721500210 91f0b04
09 – timezone (-3)

0x22 - (SPC_MAX_SPEED_LIMIT): set max speed limit
UINT8 maxSpeed: //1 Byte

Command example:
01a733637215002278fa3004
78 – maxSpeed (120Km/h)

0x23 - (SPC_DEACTIVATE_PANIC): activate or deactivate panic
No parameters or
UINT8 status: //0 deactivate; 1 activate

Command example:
01a7336372150023102175ec04
1021 – status

0x24 - (SPC_CLEAR_OLD_REPORT): clear unsent position packets log.
NO parameters

Command example:
01a733637215002441a904

0x25 - (SPC_SET_ODOMETER): set initial odometer value.
UINT32 odometer: //4 Bytes

Command example:
01a7336372150025393000000f68304
39300000 – odometer (12345meters)

0x26 - (SPC_CELL_INFO_PRESET): set if cell info is added to position packet.
UINT8 cellInfo

Command example:
01a733637215002600a10304
00 – cellInfo

0x27 - (SPC_LOW_POWER_ALERT): enable low power alert or not
UINT8 enable;

Command example:
01a73363721500271021b12004
1021 - enable

0x28 - (SPC_PANIC_MODE): set trigger panic's mode
UINT8 mode: //0-None,1-Key released, 2-Long pressed

Command example:
01a733637215002810218f3004
1021 - mode

0x29 – (SPC_ZIG_AP_POWER_LEVEL): set the zigbee's power level
UINT8 apPowerLevel: //value is 0-15

Command example:
01a73363721500290c1033d204
0c – apPowerLevel

0x2A – (SPC_ZIG_AP_RADIO_CHN): set the Zigbee's radio channel
UINT8 apRadioChan: //Value is 0-3

Command example:
01a733637215002a028e6604
02 - apRadioChan

0x2B – (SPC_ZIG_ACC_RELINK_INT): the accessories' link retry interval
UINT8 edRelinkInterval: //In minutes

Command example:
01a733637215002b0f128404
0f - edRelinkInterval

0x2C – (SPC_ZIG_ACC_CMD_INT): the sync device's command send interval
UINT8 edCmdInterval: //In hundred million seconds

Command example:
01a733637215002c05cfbc04
05 – edCmdInterval

0x2D – (SPC_ZIG_ACC_RETRY_MAX_INT): the max retry duration time device
UINT16 edTimeRetryMax: //In minutes

Command example:
01a733637215002d0a117e04
0a – edTimeRetryMax

0x2E – (SPC_ZIG_AP_REV_ALL_BC): 1 is receiving all broadcast message; 0 is only receiving the message in the list
UINT8 apRevAllBc: //1: Receive msg with User key
//1: Receive msg with Default key

Command example:
01a733637215002e024aaa04
02 – apRevAllBc

0x2F – (SPC_SET_HOURMETER): set initial hourmeter
UINT32 hourmeter: //4 Bytes

Command example:
01a733637215002f31d400006a3c04
31d4 – hourmeter (54321min)

0x30 – (SPC_SET_POSITION_GROUP): set info group of the position packets.
UINT8 infoGroup

Command example:
01a7336372150030e7bd2704
e7 – infoGroup

0x31 – (SPC_SET_SMART_OUTPUT2): enable or disable smart output port2.
UINT8 enabled: //1-enable, 0-disable

Command example:
01a73363721500311021648904
1021 – enable

0x32 – (SPC_OUTPUT_TEST): start all output test.
No parameters

Command example:
01a7336372150032b6db04

0x33 – (SPC_OUTPUT_MACRO): start output macro sequence.
No parameters

Command example:
01a733637215003397cb04

0x34 – (SPC_SET_ANTI_THEFT): enable or disable anti-theft function.
UINT8 enabled: //1 Byte

Command example:
01a73363721500341021917604
1021 – enable

0x35 – (SPC_SET_BTN_PARKING): enable or disable local parking mode
UINT8 enabled: //1 Byte

Command example:

01a73363721500351021a04504

1021 - enable

0x36 – (SPC_SET_DOOR_DETECT): enabled or disable door detect

UINT8 enabled: //1 Byte

Command example:

01a73363721500361021f3103004

1021 - enable

0x37 – (SPC_DISARMED_MODE): let device go to normal mode.

No parameters

Command example:

01a73363721500371033b04

0x38 – (SPC_PARKING_MODE): let device go to parking mode or exit.

UINT8 enter: //1 Byte

Command example:

01a73363721500381021fc3304

1021 – enter

0x39 – (SPC_ENABLE_ZIG): enable or disable zigbee communication when ignition off.

UINT8 enable: //1 Byte

Command example:

01a73363721500391021cd0004

1021 - enable

0x3A – (SPC_CALC_ODOMETER): enable or disable calculate odometer when ignition off.

UINT8 enable: //1 Byte

Command example:

01a733637215003a10219e5504

1021 - enable

0x3B – (SPC_BUZZER_CTRL): control the buzzer.

UINT8 circle

UINT8 onTime

UINT8 offTime

Command example:
01a833aaab24003b031e0557ff04
03 – cicle
1e – onTime
05 – offTime

0x3C – (SPC_ENABLE_MICROPHONE): enable or disable the microphone, MXT1xx only accept this command through USB.

UINT8 enable

Command example:
01a733637215003c1021f12204
1021 – enable

0x3D – (SPC_MOVING_TRI_ALARM): activate moving trigger alarm mode.
UINT8 enable

Command example:
01a733637215003d1021c11504
1021 – enable

0x3E – (SPC_SET_EVENT_FLAG): set the event selection

U8 selectionList[16]: Each event use one bit to enable/disable, 0 is enable, 1 is disable

Bit1: //Device power on
Bit2: //GPRS first attached or reattached
Bit3: //Transmission interval stopped
Bit4: //Transmission interval moving
Bit5: //Transmission interval in panic
Bit6: //Some configuration changed (change transmission interval or modify position packet content)
Bit7: //Server's requirement
Bit8: //Get GPS valid after transmission interval (on transmission interval the GPS does not fix)
Bit9: //Ignition on
Bit10: //Ignition off
Bit11: //Panic activated
Bit12: //Panic deactivated
Bit13: //Input 1 activated

Bit14: //Input 1 opened
Bit15: //Input 2 activated
Bit16: //Input 2 opened
Bit17: //Input 3 activated
Bit18: //Input 3 opened
Bit19: //Input 4 activated
Bit20: //Input 4 opened
Bit21: //Moving detect
Bit22: //Stopped detect
Bit23: //Anti-theft alarmed
Bit24: //At least one accessories critical
Bit25: //External power fail
Bit26: //External power ok
Bit27: //GPS antenna fail
Bit28: //GPS antenna OK
Bit29: //2.4Ghz packet received
Bit30: //Entering sleep
Bit31: //Output 1 activated
Bit32: //Output 1 deactivated
Bit33: //Output 2 activated
Bit34: //Output 2 deactivated
Bit35: //Output 3 activated
Bit36: //Output 3 deactivated
Bit37: //Maximum speed exceeded
Bit38: //Maximum speed OK (after a exceed event)
Bit39: //Entering waypoint
Bit40: //Leaving waypoint
Bit41: //Backup battery fail
Bit42: //Backup battery OK (after fail event)
Bit43: //Delivery fail
Bit44: //Require from SMS
Bit45: //Tampering is open
Bit46: //G-sensor rolling threshold reached
Bit47: //G-sensor side threshold reached
Bit48: //G-sensor shock threshold reached
Bit49: //GPS direction changed
Bit50: //SMS interval (without GPRS connection)
Bit51: //Power off
Bit52: //anti-theft enter normal from alarmed
Bit53: //GSM Jamming switches from NO to YES
Bit54: //GSM Jamming switches from YES to NO

Bit55: //Excessive RPM
Bit56: //Excessive RPM on neutral
Bit57: //Speeding on neutral
Bit58: //GPS Failure
Bit59: //distance attached
Bit60: //Power Fail and GPS Fail
Bit61: //AGPS request
Bit62: //TAG accessories status changed
Bit63: //TAG accessories battery status changed
Bit64: //Link broken
Bit65: //Expand Input changed
Bit66: //TAG accessories status changed back
Bit67: //Only have 30% power in battery
Bit68: //Only have 20% power in battery
Bit69: //Keep stopped with ignition on status
Bit70: //Improper moving
Bit71: //Camera blind
Bit72: //Camera blind recover
Bit73: //Camera video lost
Bit74: //Camera video ok
Bit75: //RS232 data incoming
Bit76: //Calibrate ignition voltage finished
Bit77: //Enter deep sleep
Bit78: //Exceed max speed in raining
Bit79: //Resume speed after exceed in raining
Bit80: //Acceleration exceed
Bit81: //Acceleration resume after exceed
Bit82: //Deceleration exceed
Bit83: //Deceleration resume after exceed
Bit84: //RFID driver login
Bit85: //RFID driver logout
Bit86: //RFID passenger login
Bit87: //Generic exceed max speed
Bit88: //Generic resume speed after exceed
Bit89: //Fail try device password more than 3 times
Bit90: //Receive engine seal activate command
Bit91: //Engine seal activated
Bit92: //Engine seal deactivated
Bit93: //Engine seal activated by relay
Bit94: //Engine seal deactivated relay
Bit95: //Engine seal activated by input1

Bit96: //Engine seal deactivated by input1
Bit97: //Network scan response
Bit 98: //Histogram of Speed
Bit 99: //Delta of Journey
Bit 100: //Telemetry events
Bit101: //Event's reconstruction(reserved)
Bit102: //Route's reconstruction
Bit103 ~ Bit128: //reserved, please keep it to "0"

Command example:
01a733637215003e02c0000000e4f91fd093c7ffffff00003ecc04
02c0000000e4f91fd093c7ffffff0000 – selectionList

0x3F – (SPC_SET_DEBOUNCE_TIMER): set the debounce timer
ignDebTimer: //1Byte, the debounce timer for ignition, in seconds.
panicDebTimer: //1Byte, the debounce timer for panic, in seconds.
ipt1DebTimer: //1Byte, the debounce timer for input1, in seconds.
ipt2DebTimer: //1Byte, the debounce timer for input2, in seconds.
ipt3DebTimer: //1Byte, the debounce timer for input3, in seconds.
ipt4DebTimer: //1Byte, the debounce timer for input4, in seconds.
maxSpeedDebTimer: //1Byte, the debounce timer for max speed, in seconds.
wpDebTimer: //1Byte, the debounce timer for waypoint, in seconds.
exPowerDebTimer: //1Byte, the debounce timer for external power detect, in seconds.

Command example:
01a7336372150003f02000506070008090a05e304
02 – ignDebTimer (2sec)
00 – panicDebTimer (0sec)
05 – ipt1DebTimer (5sec)
06 – ipt2DebTimer (6sec)
07 – ipt3DebTimer (7sec)
00 – ipt4DebTimer (0sec)
08 – maxSpeedDebTimer (8sec)
09 – wpDebTimer (9sec)
0a – exPowerDebTimer (10sec)

0x40 – (SPC_SET_OPT_MASK): set output mask flags.
opt1Mask: //1Byte, mask output1
opt2Mask: //1Byte, mask output2

opt3Mask: //1Byte, mask output3

Command example:
01a733637215004001000 1c2a904
01 – opt1Mask
00 – opt2Mask
01 – opt3Mask

0x41 – (SPC_OLD_POS_TRANS_COUNT): set the max count of the old position can be transmitted
oldPosTransCount: //2bytes, set the max count of old position will be sent after reconnected, must be less than 10001

Command example:
01a7336372150041942610331f04
9426 – oldPosTransCount (9876 positions)

0x42 – (SPC_REQ_OLD_POSITION): require the specific packets in the log.
latest position count: //2bytes, the latest packet's position count which transmit
number of transmit: //2bytes, require packets number

Command replaced by new command with MT: 0x3c (see section 3.16)

0x43 – (SPC_SET_BACKDOOR_FUNC): set the backdoor trigger the device
UINT8 backdoor: //1Byte, 0-reset, 1-power off

Command example:
01a833aaab24004310218c0d04
1021 – backdoor

0x44 – (SPC_SET_IGN_VOLTAGE): set the voltage threshold ignition measure
UINT8 threshold: //1Byte

Command example:
01a733637215004419f1ed04
19 – threshold

0x45 – (SPC_SET_OUTPUT1_INVERT): set output1 invert
UINT8 invert: //1Byte

Command example:

01a73363721500451021f94d04

1021 - invert

0x46 – (SPC_OUTPUT_MACRO2): send output macro2 command,

UINT8 output2ActivateTimer: //1Byte

UINT8 output3ActivateTimer: //1Byte

Command example:

01a73363721500461e281e8f04

1e – output2ActivateTimer

28 – output3ActivateTimer

0x47 – (SPC_CHARGING_ALLOW): enable or disable charging when ignition off

UINT8 enabled: //1Byte, 0-disable, 1-enable

Command example:

01a733637215004710219b2b04

1021 - enabled

0x48 – (SPC_GPS_ACCELERATE_FILTER): set the accelerate filter

UINT8 accfilter: //1Byte

Command example:

01a7336372150048199ca804

19 – accfilter (25Km/h/s)

0x49 – (SPC_SET_SPEAKER_ENABLE): set the speaker enable or disable

UINT8 speakerEnable: //1Byte, 0-disabled, 1-enabled

Command example:

01a833aaab2400490066f204

00 – speakerEnable

0x4A – (SPC_SET_ODORPM_ENABLE): set the inputs enable or disable for odometer and RPM measuring.

UINT8 odo_rpmEnable: //0:disable both, 1 enable odometer input only, 2 enable RPM input only, 3 enable both

Command example:

01a733637215004a1021c75d04

1021 – odo_rpmEnable

0x4B – (SPC_SET_RPM_THRESHOLD): set the threshold for RPM
UINT16 rpmThreshold: //2Bytes, in hundreds per minute

Command example:
01a733637215004b1400ee8704
1400 – rpmThreshold (2000rpm)

0x4C – (SPC_SET_ODOMETER_PULSES): set the pulses per km for odometer
UINT16 odoPulses: //2Bytes, pulses per Km

Command example:
01a733637215004c4c1df44604
4c1d – odoPulses (7500 pulses/Km

0x4D – (SPC_SET_RPM_PULSES): set the pulses per revolution for RPM
UINT16 rpmPulses: //2Bytes, pulses per Rotation

Command example:
01a733637215004d0a00321504
0a00 – rpmPulses (10 pulses/rpm)

0x4E – (SPC_GSR_RS_MODE): set MXT1xx position inside vehicle (rolling axis/side axis):
UINT8 mode: //0 - none,
//1 - X/Y,
//2 - X/Z,
//3 - Y/X,
//4 - Y/Z,
//5 - Z/X,
//6 - Z/Y
//7 – auto configure

Command example:
01a733637215004e0341b104
03 – mode (Y/X)

0x4F – (SPC_GSR_ROLLING_THRESHOLD): set rolling axis threshold
UINT8 threshold: //1Byte(00~0F), in G's/10

Command example:
01a733637215004f0c9f7304

0c - threshold

0x50 – (SPC_GSR_SIDE_THRESHOLD): set side axis threshold
UINT8 threshold: //1Byte(00~0F), in G's/10

Command example:
01a73363721500500df37004
0d – threshold

0x51 – (SPC_GSR_SHOCK_THRESHOLD): set shock axis threshold
UINT8 threshold: //1Byte(00~0F), in G's/2

Command example:
01a73363721500510b10242304
0b – threshold

0x52 – (SPC_INCOMING_BUZZER): enable buzzer when incoming call
UINT8 enable: //0-disable, 1-enable

Command example:
01a733637215005210219ba204
1021 – enable

0x53 – (SPC_SMS_SEND_COUNT): set max count sending SMS when no gprs.
UINT8 maxCount: //1Byte

Command example:
01a7336372150053d2321f04
d2 – maxCount

0x54 – (SPC_SMS_SENDING_INTERVAL): set the interval in seconds sending through SMS
UINT16 interval: //2Bytes, in seconds

Command example:
01a7336372150054393055ae04
3930 – interval

0x55 – (SPC_SMS_PANIC_NUMBER): set the number of sending SMS when entering panic or alerting status.
UINT8 index: //the index of the three number's, start from 1
UINT8 numSize //the size of the number, if the size = 0 means remove

		this number
UINT8*	number	//numSize Bytes

Command example:

01a7336372150055020a3331383838383434345d9d04

02 – index

0a – numSize

3331383838383434343 – number (ASCII value)

0x56 – (SPC_CFG_ALIAS_NAME): set configuration alias name

UINT8	aliasSize:	//the size of the alias name
UINT8	alias:	//aliasSize Bytes

Command example:

01a7336372150056084d6178747261636b06b404

08 – aliasSize

4d6178747261636b – alias

0x57 – (SPC_OUTPUT_MACRO3):

UINT8	index:	//1byte, index of the output, start from 1
UINT16	ActivationDuration:	//2bytes, seconds, 0xFFFF means continuous, 0x0000 means skip activation
UINT16	DeactivationDuration:	//2bytes, seconds, 0xFFFF means continuous, 0x0000 means skip deactivation
UINT16	cycles:	//2bytes, cycles of on/off, 0xFFFF means continuous, 0x0000 has no meaning, same as 0x0001

Command example:

01a7336372150057102102000a000300d3b704

1021 – index (out1)

0200 – ActivationDuration

0a00 – DeactivationDuration

0300 – cycles

0x58 – (SPC_KEEP_WORKING_BF_SLEEP):

UINT8	keepworkingtimer:	//1byte, in hours

Command example:

01a7336372150058807fb904

80 – keepworkingtimer

0x59 – (SPC_SET_SPEAKER_VOL):

UINT8	speakerVol:	//1byte, from 1 to 10

Command example:

01a833aaab24005907f28104

07 - speaker

0x5A – (SPC_SET_CALL_NUMBER):

UINT8	mode:	//1byte, 0 is set outgoing call; 1 is set incoming call
UINT8	idx:	//1byte, the index of call list, start from 0
UINT8	numSize:	//1byte, the number size, if size<3 means remove this number, if size >=20 set fail.
	number:	//numSize bytes, the number.

Command example:

01a4335ae416005a1021020a33313636363637373737d35c04

1021 – mode

02 – idx

0a – numSize

333313636363637373737 – number

0x5B – (SPC_SET_MAKE_CALL_TIME):

UINT8	makeCallTime:	//1byte, set the outgoing call during time in minutes;

Command example:

01a4335ae416005b4189fa04

41 – makeCallTime (65min)

0x5C – (SPC_SET_RING_MODE):

UINT8	ringMode:	//1byte, set ring mode when incoming call //0 – disable warning //1 – only ring //2 – only vibrate //3 – both ring and vibrate

Command example:

01a4335ae416005c02b91b04

02 – ringMode

0x5D – (SPC_UPD_BY_ZIGBEE):

UINT8	enable:	//0-disable, 1-enable

Command example:
01a733637215005d1021aa8e04
1021 – enable

0x5E – (SPC_SET_ZIG_ALERT_TIME): set zigbee alert interval
UINT8 zigAlarmTime: //1Byte, in seconds

Command example:
01a733637215005eb40e6504
b4 - zigAlarmTime

0x5F – (SPC_SET_ZIG_ALERT_INFO): set zigbee alert information
UINT8 infoSize
UINT8 info[infoSize]

Command example:
01a733637215005f084d4158545241434b4e5104
08 – infoSize
4d4158545241434b – info (ASCII value)

0x60 – (SPC_SET_TIMEOUT_NOGSM): set the timeout of no GSM signal
UINT8 timeWithoutGsm: //in minutes

Command example:
01a733637215006087a44504
87 – timeWithoutGsm

0x61 – (SPC_SET_TIMEOUT_MOVING): set the timeout moving
UINT16 timeoutMoving: //in minutes

Command example:
01a733637215006110300ee3eb04
10300e – timeoutMoving

0x62 – (SPC_ACTIVATE_ZIG_ALERT): activate or deactivate zigbee alert
UINT8 alertTime: //in seconds

Command example:
01a733637215006202ebe204
02 – alertTime

0x63 – (SPC_SET_ZIG_RSSI_FILTER): set the zigbee RSSI filter
UINT8 zigRssiFilter: //from 0 to 15

Command example:
01a73363721500630c143004
0c – zigRssiFilter

0x64 – (SPC_SET_IGNITION_CODE): set the ignition code
UINT8 ignCode: //up to 99

Command example:
01a7336372150064547e7204
54 – ignCode

0x65 – (SPC_INPUT_ON_PANIC_ANTI): set the key and door as input
UINT8 bBtnDoorAsInput: //0-disable, 1-enable

Command example:
01a733637215006510211f4b04
1021 – bBtnDoorAsInput

0x66 – (SPC_ENABLE_GPS_FILTER): enable or disable GPS filter
UINT8 gpsFilter: //0-disable, 1-enable

Command example:
01a733637215006610214c1e04
1021 – gpsFilter

0x67 – (SPC_SET_AGPS_TIMER): set AGPS interval
UINT8 timer: //1Byte

Command example:
01a7336372150067235d2904
23 – timer

0x68 – (SPC_DEV_LINK_TIMER): set link interval with others device
UINT8 timer: //in seconds

Command example:
01a7336372150068af676904

af – timer

0x69 – (SPC_NEUTRAL_SPEED_RPM): Speeding on Neutral
UINT8 neutralMaxSpeed: //in Km/h
UINT16 neutralRpm: //in hundreds per minute

Command example:
01a73363721500694b09002e1d04
4b – neutralMaxSpeed
0900 – neutralRpm

0x6A – (SPC_NEUTRAL_EXCESSIVE_RPM): Excessive RPM on Neutral
UINT16 neutralExcessiveRpm: //Excessive RPM stopped

Command example:
01a733637215006a3800937504
3800 - neutralExcessiveRpm

0x6B – (SPC_GPS_FAILURE_TIMER): set GPS failure debounce timer
UINT8 timer: //in minutes

Command example:
01a733637215006b5a8e8304
5a – timer

0x6C – (SPC_SET_EXCEEDSPEED_OUTPUT): set which output will be activated when exceed max speed.
UINT8 whichOutput

Command example:
01a733637215006c02e4c104
02 – whichOutput

0x6D – (SPC_SET_LINKFAIL_OUTPUT): set which output will be activated when link other device failed.
UINT8 whichOutput
UINT8 linkFailTimes

Command example:
01a733637215006d0208559b04
02 – whichOutput

08 – linkFailTimes

0x6E – (SPC_SET_JAMMING_OUTPUT): set which output will be activated when GSM jamming

UINT8 whichOutput

Command example:

01a733637215006e1021e59704

1021 – whichOutput

0x6F – (SPC_SET_JAMMING_ALERT): set alert through 2.4Ghz when Jamming

UINT8 enable: //1-enable, 0-disable

Command example:

01a733637215006f1021d4a404

1021 – enable

0x70 – (SPC_SET_SIREN): set siren working when anti-theft alarmed.

UINT8 activateDuration: //in seconds

UINT8 deactivateDuration: //in seconds

UINT8 circle: //cycles of siren

UINT8 output: //1 is output2; 2 is output 3

Command example:

01a73363721500703c0a6410217ddc04

3c – activateDuration

0a – deactivateDuration

64 – circle

1021 – output

0x71 – (SPC_SET_DISTANCE_THRESHOLD):

UINT8 distanceThreshold: //in hundred meters

Command example:

01a7336372150071 96f67704

96 – distanceThreshold

0x72 – (SPC_SET_DIRECTION_THRESHOLD):

UINT8 directionThreshold: //from 10 to 180 degrees

Command example:

01a733637215007278453e04

78 – directionThreshold

0x73 – (SPC_SET_IP_PRIORITY):

UINT8	ipPriority:	//0: None //1: IP 1 //2: IP 2

Command example:

01a733637215007302a9d204

02 – ipPriority

0x74 – (SPC_PACKET_ENCRYPT_KEY): encrypt key for all positions packets

UINT8	packetEncryptKey[16]:	//

Command example:

01a7336372150074102102031024050607080 90a0b0c0d0e0f66806e04

102102031024050607080 90a0b0c0d0e0f66 – packetEncryptKey

0x75 – (SPC_APN2): set secondary apn

UINT8	apn2Size:	//1byte, apn2 size
UINT8	user2Size:	//1byte, user2 name size
UINT8	pwd2Size:	//1byte, password2 size
UINT8	apn2:	//apn2Size bytes, the apn2
UINT8	user2:	//user2Size bytes, the user2 name
UINT8	password2	//pwd2Size bytes, the password2

Command example:

01a733637215007 50c0505636c61726f2e636f6d2e6272636c61726f636c61726ffc6904

0c – apn2Size

05 – user2Size

05 – user2Size

636c61726f2e636f6d2e6272 – apn2

636c61726f – user2

636c61726f – password2

0x76 – (SPC_SET_RS_DEBOUNCE): set G-Sensor RS debounce

UINT8	rsRollingDeb:	//1Byte, in seconds
UINT8	rsSideDeb:	//1Byte, in seconds
UINT8	rsShockDeb:	//1Byte, in seconds

Command example:
01a733637215007619232d2bfe04
19 – rsRollingDeb
23 – rsSideDeb
2d – rsShockDeb

0x77 – (SPC_ANTI_ALERT_TIMER): set anti-theft alarmed mode timer
UINT8 alertTimer: //1byte, in seconds

Command example:
01a7336372150077149a6c04
14 – alertTimer

0x78 – (SPC_SET_RS_OUTPUT): set G-Sensor RS output
UINT8 rollingOpt: //1Byte
UINT8 sideOpt: //1Byte
UINT8 shockOpt: //1Byte

Command example:
01a7336372150078102102102 18a6604
1021 – rollingOpt
02 – sideOpt
1021 – shockOpt

0x79 – (SPC_SET_PARKING_OPT): set anti-theft parking output
UINT8 parkingOpt: //1byte

Command example:
01a733637215007902623d04
02 – parkingOpt

0x7A – (SPC_SET_GAO_SPEED):
UINT8 bGandOSpeed: //1byte

Command example:
01a733637215007a00734804
00 – bGandOSpeed

0x7B – (SPC_SET_SENDING_ORDER):
UINT8 bGrowingSending: //1Byte, 1-growing, 0- descending

Command example:

01a733637215007b1021636b04

1021 – bGrowningSending

0x7C – (SPC_ENABLE_CHG_NET_SET): allow set apn by iMXT

UINT8 bModifyApn: //1Byte, 1-enable, 0 disable

Command example:

01a733637215007c10215c3f04

1021 – bModifyApn

0x7D – (SPC_SET_ZIG_TAG_INTERVAL): set zig tag analisys timer

UINT16 zigTagInterval: //2bytes, in seconds

Command example:

01a733637215007d1700b8a504

1700 – zigTagInterval

0x7E – (SPC_EX_OUTPUT_CTRL)

UINT8 wt200 index: //1Byte

UINT8 output index: //1Byte

UINT8 activate: //1Byte

Command example:

01a733637215007e1021031021664704

1021 – wt200index

03 – outputindex

1021 – activate

0x7F – (SPC_EX_INPUT_MASK):

UINT8 wt200 index : 1byte

UINT8 mask : 1byte

Command example:

01a733637215007f10211021305704

1021 – wt200index

1021 – mask

0x80 – (SPC_SET_PANIC_VIA): set which input to activate panic

UINT8 panicVia: //1: Input1

//2: Wireless Button
//3: Both

Command example:
01a7336372150080023b9404
02 – panicVia

0x81 – (SPC_SET_TAG_FAIL_OUTPUT):
UINT8 tagFailOpt: //0: None
//1: Output1
//2: Output2
//3: Output3

Command example:
01a73363721500811021699704
1021 – tagFailOpt

0x82 – (SPC_ENABLE_AP_SET_OUTPUT):
UINT8 bApOptAllow: //1byte, 0-disable, 1-enable

Command example:
01a73363721500821021 0fc704
1021 – tagFailOpt

0x83 – (SPC_SET_RPM_DEBOUNCE):
UINT8 debSpeedNeutral: //1byte, from 0 to 255
UINT8 debMaxExcessiveRpm: //1byte, from 0 to 255
UINT8 debMaxExcessiveRpmNeutral: //1byte, from 0 to 255

Command example:
01a73363721500831a1b1c3d2e04
1a – debSpeedNeutral
1b – debMaxExcessiveRpm
1c – debMaxExcessiveRpmNeutral

0x84 – (SPC_SET_PARKING_OPT_MODE):
UINT8 parkingOptAct: //1byte, from 0 to 255
UINT8 parkingOptDeact: //1byte, from 0 to 255
UINT8 parkingOptCircle: //1byte, from 0 to 255
UINT8 parkingOptInterval //1byte, from 0 to 255

Command example:

01a73363721500848c0e6414a80504

8c – parkingOptAct

0e – parkingOptDeact

64 – parkingOptCircle

14 – parkingOptInterval

0x85 – (SPC_SET_GPS_FOR_ALERT):

UINT8 bGpsForAlert: //1byte, from 0 to 1

Command example:

01a73363721500851021ad5b04

1021 – bGpsForAlert

0x86 – (SPC_SET_RPM_FACTOR):

UINT8 rpmFactor: //0: multiplied by 1
//1: multiplied by 10
//2: multiplied by 100
//3: multiplied by 1000

Command example:

01a7336372150086029d3e04

02 – rpmFactor

0x87 – (SPC_SET_GSR_DEBFACTOR):

UINT8 gsrDebFactor: //0: multiplied by 1
//1: multiplied by 10
//2: multiplied by 100

Command example:

01a73363721500871021cf3d04

1021 – gsrDebFactor

0x88 – (SPC_SET_RPM_TRIGGER):

UINT8 rpmUpTrigger: //1byte, from 0 to 1

Command example:

01a73363721500881021ce0004

1021 – rpmUpTrigger

0x89 – (SPC_SMS_SPEED_ALERT_NUM):

UINT8 numSize: //1byte, the number size
UINT8 number[]: //numSize bytes, the number

Command example:
01a73363721500890a3331353535353532323232d6e604
0a – numSize
33313535353532323232 – number (ASCII value)

0x8A – (SPC_RPM_EVENT_OUTPUT):
UINT8 rpmEvt1Opt: //1byte, from 0~3
UINT8 rpmEvt2Opt: //1byte, from 0~3
UINT8 rpmEvt3Opt: //1byte, from 0~3

Command example:
01a733637215008a00102102af4604
00 – rpmEvt1Opt
1021 – rpmEvt2Opt
02 – rpmEvt03Opt

0x8B – (SPC_IMPROPER_MOVING_OUTPUT):
UINT8 improperMovingOpt: //1byte, from 0~3

Command example:
01a733637215008b02c14804
02 – improperMovingOpt

0x8C – (SPC_ALLOW_ANY_TAG_ACC):
UINT8 anyWt110: //1byte, 0 to 1

Command example:
01a733637215008c102135e104
1021 – anyWt110

0x8D – (SPC_SET_LONG_TIMER_NOMOVING):
UINT8 ignLongTimer: //1byte, in minutes

Command example:
01a733637215008d7d1f6d04
7d – ignLongTimer

0x8E – (SPC_GSR_THRESHOLD_LEVEL): set G-Sensor detec level

UINT8 gsrLevel: 1byte

Command example:
01a733637215008e00769704
00 – gsrLevel

0x8F – (SPC_SET_SIM_CHIP): set if device will use SimChip
UINT8 simChip: //1byte

Command example:
01a733637215008f102166b404
1021 – simChip

0x90 – (SPC_ZIG_WORKING_MODE):
UINT8 exWorkingMode: //0: Normal
//1: As WT110
//2: Disable RF

Command example:
01a733637215009002489704
02 – exWorkingMode

0x91 – (SPC_SET_IGN_VOL_FACTOR):
UINT8 factor: //0: factor 1
//1: factor 2
//2: factor 5
//3: factor 10

Command example:
01a733637215009102 79a404
02 – factor

0x92 – (SPC_CALIBRATE_IGN_VOLTS):
No Parameter

Command example:
01a73363721500925c6e04

0x93 – (SPC_SET_STOP_INTERVAL_FACTOR):
UINT8 factor: //0: factor 1
//1: factor 10

//2: factor 100

Command example:
01a7336372150093021bc204
02 – factor

0x94 – (SPC_SET_INPUT1_ACTION):
UINT8 input1Action: //1byte, 1-enable, 0-disable

Command example:
01a73363721500941021ef6b04
1021 – input1Action

0x95 – (SPC_SET_RS232_WORKING_MODE):
UINT8 rs232WorkingMode: //0: Disable
//1: Enable when ignition on
//2: Enable all time
//3: RFID Driver
//4: RFID Passenger
//5: Taximeter

Command example:
01a7336372150095039c7804
03 – rs232WorkingMode

0x96 – (SPC_SET_RS232_WORKING_TIMER):
UINT8 rs232WorkingTimer: //1byte

Command example:
01a7336372150096b4f3fa04
b4 – rs232WorkingTimer

0x97 – (SPC_SET_RS232_SPEED):
UINT8 rs232Speed: //0: 9600bps
//1: 19200bps
//2: 38400bps
//3: 57600bps
//4: 115200bps

Command example:
01a733637215009703fe1e04

03 – rs232Speed

0x98 – (SPC_SET_RS232_RECV_TIMEOUT):
UINT8 rs232RecvTimeout: //1byte, from 20 to 255, multiplied by 10

Command example:
01a7336372150098 96dcdd04
96 – rs232RecvTimeout

0x99 – (SPC_SET_RS232_ALERT):
UINT8 rs232Alert: //1byte, 1-enable, 0-disable

Command example:
01a73363721500991021b31d04
1021 – rs232Alert

0x9A – (SPC_ANTI_ALARM_AFTER_TIMER):
UINT8 alarmAfterTimer: //1byte, 1-enable, 0-disable

Command example:
01a733637215009a1021e04804
1021 – alarmAfterTimer

0x9B – (SPC_ANTI_STATUS_INFO):
UINT8 antiStatInfo: //1byte, 1-enable, 0-disable

Command example:
01a733637215009b1021d17b04
1021 – antiStatInfo

0x9C – (SPC_ANTI_SILENCE_LAW):
UINT8 antiSilence: //1byte, 1-enable, 0-disable

Command example:
01a733637215009c102146e204
1021 – antiSilence

0x9D – (SPC_SET_RAIN_INPUT):
UINT8 rainInput: //0: None
//1: Input 1
//2: Input 2

//3: Input 3

Command example:
01a733637215009d0214e104
02 - rainInput

0x9E – (SPC_SET_RAIN_MAXSPEED):
UINT8 rainMaxSpeed: //1byte, in Km/h

Command example:
01a733637215009e789a6b04
78 – rainMaxSpeed

0x9F – (SPC_UDP_BIND_PORT): set keep udp port
UINT8 bindPort: //1byte, 1-enable, 0-disable

Command example:
01a733637215009f102115b704
1021 – bindPort

0xA0 – (SPC_SET_ACCELERATION):
UINT8 accelerationOutput: //0:None, 1:Output1, 2:Output2
UINT8 accelerationLimit: //1byte

Command example:
01a73363721500a0024b548704
02 – accelerationOutput
4b – accelerationLimit

0xA1 – (SPC_SET_DECELERATION):
UINT8 decelerationOutput: //0:None, 1:Output1, 2:Output2
UINT8 decelerationLimit: //1byte

Command example:
01a73363721500a11021166f6e04
1021 – decelerationOutput
16 – decelerationLimit

0xA2 – (SPC_RFID_ANY_DEVICE):
UINT8 bAnyRfid: //1byte, 1-enable, 0-disable

Command example:

01a73363721500a21021dcc404

1021 – bAnyRfid

0xA3 – (SPC_RFID_LOGOUT_MODE):

UINT8 rfidLogoutMode: //0: RFID Card, 1: Ignition Off

Command example:

01a73363721500a31021edf704

1021 – rfidLogoutMode

0xA4 – (SPC_RFID_PASS_OPT):

UINT8 rfidPassengerOpt: //1byte, 1-true, 0-false

Command example:

01a73363721500a410217a6e04

1021 – rfidPassengerOpt

0xA5 – (SPC_SET_TIMEZONE_EX):

SINT16 minutes: //2bytes, -720~780

Command example:

01a73363721500a50ea4ac04

0e – minutes

0xA6 – (SPC_SET_INPUT_INVERT):

UINT8 input: //1byte

UINT8 invert: //1byte

Command example:

01a73363721500a6021021a603102129bd04

02 – input

1021 – invert

03 – input

1021 – invert

0xA7 – (SPC_SET_RAINSENSOR_SIGNAL):

UINT8 rainSensorSignal: //0: Pulses, 1: Continuos

Command example:

01a73363721500a71021293b04

1021 – rainSensorSignal

0xA8 – (SPC_SET_GENERIC_EVENT):

UINT8	input:	//1byte, from 0~3
UINT8	output:	//1byte, from 0~2
UINT8	maxspeed:	//1byte

Command example:

01a73363721500a803102182515404

03 – input

1021 – output

82 – maxspeed

0xA9 – (SPC_SET_SMART_OUTPUT1):

UINT8 smartOutput1: 1byte

Command example:

01a73363721500a9102138b104

1021 – smartOutput1

0xAA – (SPC_SMS_ENABLE_ACK):

UINT8 smsAckEnabled: 1byte

Command example:

01a73363721500aa102168e804

1021 – smsAckEnabled

0xAB – (SPC_ENGINE_SEAL_OPT):

UINT8	engineSealOpt:	//0-None
		//1-Output1
		//2-Output2
UINT8	engineSealCondition:	//1byte (0~7), 1: Ignition Off
		//1: G-Sensor Stopped
		//1: GPS Speed = 0Km/h
UINT8 engineSealOptMode :		//0-Immediate
		//1-Custom Macro

Command example:

01a633189a7b00ab02021021165104

02 – engineSealOpt

02 – enginseSealCondition

1021 – engineSealOptMode

0xAC – (SPC_ENGINE_SEAL_RELAY):
UINT8 engineSealRelay: //1byte, 1-true, 0-false
UINT8 engineSealDeb: //1byte

Command example:
01a633189a7b00ac10217835d504
1021 – engineSealRelay
78 – engineSealDeb

0xAD – (SPC_ENGINE_SEAL_JAMMMODE):
UINT8 engineSealJammMode: //1byte, 1-true, 0-false

Command example:
01a633189a7b00ad1021a6d704
1021 – engineSealJammMode

0xAE – (SPC_ENGINE_SEAL_ACTIVATE):
UINT8 activated: //1byte, 1-activated, 0-deactivated

Command example:
01a633189a7b00ae1021f58204
1021 – activate

0xAF – (SPC_CAL_IGN_VOL_OUTPUT): set auto ignition threshold
UINT8 calIgnVolOutput: //0-None
//1-Output1
//2-Output2
Command example:
01a633189a7b00af1021c4b104
1021 – calIgnVolOutput

0xB0 – (SPC_ENGINE_SEAL_INPUT):
UINT8 engineSealInput: //1byte, 1-true, 0-false

Command example:
01a633189a7b00b0102189a204
1021 – engineSealInput

0xB1 – (SPC_SENDING_INTERVAL_EX):

UINT16 stopInterval:
UINT16 moveInterval

0xB2 – (SPC_ENGINE_SEAL_OPT_CYCLE):
UINT8 engineSealOptCycle:
UINT8 esocActMinutes
UINT8 esocDeaMinutes

0xB3 – (SPC_AGPS_ENABLE):
UINT8 agpsEnable 1 Byte
// 0x0 – Disabled;
// 0x1 – Enabled;

Command example:
// Disable AGPS
01ad33b7a1b800b30016dd04
// Enable AGPS
01ad33b7a1b800b3102137cd04

0xB4 – (SPC_AGPS_SERVER) ***Not available****;*
UINT8 userSize
UINT8 passwordSize;
UINT8 urlSize
UINT16 port
UINT8 user[]
UINT8 password[]
UINT8 url[]

Command example:
01ad33b7a1b800B413050f62b56D616C5F77616E67406D6F7273756E2E636F6D51616C656A616770732E752D626C6F782E636F6DFE5004

0xB5 – (SPC_CELL_INFO_EX)
UINT8 lbsType 1 Byte
// 0x0 – Disabled LBS;
// 0x1 – Enabled only when GPS not fixed;
// 0x2 – Enabled always.

Command example:
// Disabled LBS
01ad33b7a1b800b500b07704
// Enabled only when GPS not fixed;

01ad33b7a1b800b51021916704

// Enabled always
01ad33b7a1b800b502f25704

0xB6 – (SPC_LBS_REQUEST)
No Parameters

Command example:
01ad33b7a1b800b6028b04

0xB7 – (SPC_PHY_IGN_PIN):
UINT8 phyIgnitionPin: 1byte [0:Prioritize ignition virtual ; 1:Prioritize ignition physical pin]

Command example:
01A73360EC7E00B71021555804

0xB8 – (SPC_KEEPALIVE_CLOUD):
UINT16 keepaliveInterval: 2 bytes [0, 3 to 1440 minutes]

Command example:
01A73360EC7E00B80300AEC404

0xB9 – (SPC_GSR_FOR_IGNVOL):
UINT8 gsrForIgnVol: 1byte [0:only ignition logic ; 1:ignition logic and accelerometer logic]

Command example:
01A73360EC7E00B910215A7B04

0xBA – (SPC_OPT_MODE_FOR_ACTION):
UINT8 optModeForAction: 1byte [0:immediate; 1:progressive]

Command example:
01A73360EC7E00BA1021092E04

0xBB – (SPC_OPT_FRE_DURING_DEB):
UINT8 optFreDuringDeb: 1byte [0:immediate; 1:means 1Hz; 2:means 2Hz; 4:means 4Hz]

Command example:
01A73360EC7E00BB025B2D04

0xBC – (SPC_SET_ACCEL_ALARM):
UINT16 accelGEnter: 2bytes

Command example:
01AC33226DAC00BCFA0062EA04

0XBD – (SPC_SET_ACCEL_THRESHOLD):
UINT16 accelThreshold: 2bytes

Command example:
01AC33226DAC00BD2003DD1704

0xBE – (SPC_SET_DECEL_ALARM):
UINT16 decelGEnter: 2bytes

Command example:
01AC33226DAC00BEF410212CB704

0xBF – (SPC_SET_DECEL_THRESHOLD):
UINT16 decelThreshold: 2bytes

Command example:
01AC33226DAC00BF2003BD7904

0XC0 – (SPC_SET_CURVE_ALARM):
UINT16 curveGEnter: 2bytes

Command example:
01AC33226DAC00C05E1021908604

0xC1 – (SPC_SET_CURVE_THRESHOLD):
UINT16 curveThreshold: 2bytes

Command example:
01AC33226DAC00C12003B4BA04

0xC2 – (SPC_SET_CRASH_ALARM):
UINT16 crashGEnter: 2bytes

Command example:

01AC33226DAC00C2C409D89E04

0xC3 – (SPC_SET_CRASH_THRESHOLD):
UINT16 crashThreshold: 2bytes

Command example:
01AC33226DAC00C396001C5004

0xC4 – (SPC_SET_PITCH_EVT):
SINT8 Pitching positive //1 byte [from 0 to +90]
SINT8 Pitching negative //1 byte [from -90 to 0]

0xC5 – (SPC_SET_ROLL_EVT):
SINT8 Rolling positive //1 byte [from 0 to +90]
SINT8 Rolling negative //1 byte [from -90 to 0]

0xC6 – (SPC_RESET_CAL_EVT)
No Parameters

Command example:
01AC33226DAC00C61C2B04

0xC7 – (SPC_HISTOGRAM_CONFIG)
UINT8 Number of range //1 byte [from 0 to 32]
UINT8 Limit of speed for each range //1 byte [from 0 to 255 Km/h]

Command example:
01AC33226DAC00C70F0F40DF04

0xC8 – (SPC_SET_SPEED_SUMMARY_IN_POS)
UINT8 Configure the Speed details //1 byte [0: Disable ; 1: Enable]

Command example:
01AC33226DAC00C8102127BA04

0xC9 – (SPC_SET_OBD_ENABLED)
UINT8 Enable OBD //1 byte [0: Disable ; 1: Enable]

0xCA – (SPC_SET_CAN_CFG_DATASET)
UINT16 CAN library dataset index // 2bytes

0xCB – (SPC_SET_CAN_IN_POS)
UINT8 Enable CAN additional packet //1 byte [0: Disable ; 1: Enable]

Command example:
01AC33226DAC00CB102174EF04

0xCC – (SPC_SET_CAN_POS_CFG)
Bit Map to specify which data that will be included in the position packet. //8 bytes

Rpm	//1 bit;
Speed	//1 bit;
Odometer	//1 bit;
Clutch	//1 bit;
Brake	//1 bit;
ParkingBrake	//1 bit;
MotorBrake	//1 bit;
FuelLevel1	//1 bit;
FuelLevel2	//1 bit;
EngineTemp	//1 bit;
FuelConsumption	//1 bit;
WsWipers	//1 bit;
DoorClosed	//1 bit;
DoorLocked	//1 bit;
SeatBelt	//1 bit;
Headlights	//1 bit;
Trunk	//1 bit;
IntakeAirTemp	//1 bit;
IntakeAirFlow	//1 bit;
ThrottlePosition	//1 bit;
BarometricPressure	//1 bit;
ControlModuleVoltage	//1 bit;
AirTemperature	//1 bit;
FuelType	//1 bit;
EthanolRatio	//1 bit;
OilTemperature	//1 bit;
EngineFuelRate	//1 bit;
EngineRefTorque	//1 bit;
MalfunctionIndLamp	//1 bit;
CurrentGear	//1 bit;
DTCs	//1 bit;

dummy1 //33 bits;

Command example:
01AC33226DAC00CCFFFFFEBFFFFFFFFF66D504

0xCD – (SPC_SET_CAN_ENABLE)
UINT8 Enable CAN module //1 byte [0: Disable ; 1: Enable]

Command example:
01AC33226DAC00CD1021D24504

0xCE – (SPC_SET_OUTPUT_ON_EVENT)
This command allows set the parameters to activate/deactivate the output alert for each event listed always in this sequence below. The content for each event must have the structure ***evtOutputCtrl_t*** described.

1. Hard breaking
2. Hard acceleration
3. Hard lateral trigger [right]
4. Hard lateral trigger [left]
5. Auto calibration [stage 1]
6. Auto calibration [stage 2 – complete]
7. Reset calibration or change the installation position
8. Crash event
9. Reserved
10. Reserved

typedef struct
{
UINT8 Output selected //1 byte [0:disable; 1: output1]
UINT8 Output cycles //1 byte [0~255]
UINT16 Timer to keep output activated //2 bytes [0~65535 milliseconds]
UINT16 Timer to keep output deactivated //2 bytes [0~65535 milliseconds]
} ***evtOutputCtrl_t***;

Command example:
01 AC 33 22 6D AC 00 CE 10 21 03 02 00 03 00 10 21 05 03 00 10 21 00 10 21 02 08 00 02 00 10 21 02 02 00 08 00 10 21 10 24 10 21 00 10 21 00 10 21 10 21 32 00 10 21 00 10 21 06 10 21 00 10 21 00 10 21 0A 05 00 10 21 00 00 00 00 00 00 00 00 00 00 00 00 00 7E 85 04

0xFF – (SPC_RESET_TO_DEFAULT): reset all configurations to default value.
No parameters

Command example:
01a633189a7b00ff0c1b04

Below is the parameters range and projects accepted table:

Command ID	Projects	Parameters Range
0x3 – (SPC_DTMF_PWD)	All	pwdSize: 0 or 4~8 password: pwdSize bytes
0x4 – (SPC_CONNECT_MODE)	All	type: 0(UDP) or 1(TCP)
0x5 – (SPC_APN)	All	apnSize: 0~99 userSize: 0~31 pwdSize: 0~31 apn: apnSize bytes user: userSize bytes password: pwdSize bytes
0x6 – (SPC_IP_ADDR)	IpIndex=0,1 (All) IpIndex=2 (MXT15X+ MXT120)	ipIndex: 0(primary ip) 1(secondary ip) 2(AGPS server ip) addrSize: 0~63 port: 0~65535 addr: AddrSize bytes
0x7 – (SPC_KEEP_ALIVE_TIMER)	All	keepAliveTimer: 0~65535
0x8 – (SPC_RI_STOPPED)	All	timerOfIgnitionOff: 0 or 5~65535
0x9 – (SPC_RI_MOVING)	All	timerOfMovement: 5~65535
0xA – (SPC_RI_PANIC)	All	timerOfPanic: 5~65535
0xB – (SPC_RI_RESEND)	All	attempts: 3~10 timeout: 10~60
0xC – (SPC_GSR_DEB_MOVING)	All	timer: 0~65535
0xD – (SPC_GSR_DEB_STOPPED)	All	timer: 0~65535
0xE – (SPC_GSR_DETECT)	All	timer: 5~65535
0xF – (SPC_GSR_REPORT)	All	sendImmediately: 0 or 1
0x10 – (SPC_GPS_KEEP_WORKING)	MXT10X; MXT120	timer: 0~65535
0x11 – (SPC_GPS_UNFIX_TIMEOUT)	All	timer: 60~65535
0x12– (SPC_GPS_COLDSTART_UNFIX_TIMEOUT)	All	timer: 300~65535
0x13 – (SPC_GPS_OPEN_BEF_TRANS)	All	MovingTimer and StoppedTimer: < unfix timeout (ID 0x11) and < coldstart unfix timeout (ID 0x12)

0x14 – (SPC_SMS_ALIAS)	All	aliasSize: 0~25 alias: aliasSize bytes
0x15 – (SPC_SMS_DESTINATION)	All	destSize: 0~15 destination: destSize bytes
0x16 – (SPC_SMS_SEND_MODE)	All	smsMode: 0~2 allowNumberMode: 0 or 1
0x17 – (SPC_ZIG_AP_SLEEP)	MXT101 MXT151 MXT151+	masterSleepEnable: 0 or 1
0x18 – (SPC_ZIG_KA_INTERVAL)	MXT101 MXT151 MXT151+	timer: 0~255
0x19 – (SPC_ZIG_KA_DURATION)	MXT101 MXT151 MXT151+	timer: 0~255
0x1A – (SPC_ZIG_ACC_RETRY)	MXT101 MXT151 MXT151+	accRetry: 0~3
0x1B – (SPC_ZIG_ACC_RX_TIMEOUT)	MXT101 MXT151 MXT151+	timeout: 0~60000
0x1C – (SPC_ZIG_ACC_ENCRYPT_KEY)	MXT101 MXT151 MXT151+	accEncryptKey: 16bytes key
0x1D – (SPC_LED_ENABLE)	All	Enable: 0 or 1
0x1E – (SPC_CHARGING_ONLY)	MXT10X MXT120	chargingOnly: 0 or 1
0x1F – (SPC_INPUT_ENABLE)	MXT15X MXT15X+	inputIndex: 1~4 enable: 0 or 1
0x20 – (SPC_OUPUT_ACTIVATE)	MXT15X MXT15X+	outputIndex: 1~3 activate: 0 or 1
0x21 – (SPC_TIME_ZONE)	All	timezone: 0~25
0x22 – (SPC_MAX_SPEED_LIMIT)	All	maxSpeed: 0~255
0x23 – (SPC_DEACTIVATE_PANIC)	All	No Parameters means: deactivate panic status: 0~1
0x24 – (SPC_CLEAR_OLD_REPORT)	All	No Parameters
0x25 – (SPC_SET_ODOMETER)	All	odometer: 0~ 4294967295
0x26 – (SPC_CELL_INFO_PRESET)	All	cellInfo: 0 or 1
0x27 – (SPC_LOW_POWER_ALERT)	MXT10X MXT120	enable: 0 or 1

0x28 – (SPC_PANIC_MODE)	All	mode:	0~2
0x29 – (SPC_ZIG_AP_POWER_LEVEL)	MXT101 MXT151 MXT151+	apPowerLevel:	0~15
0x2A – (SPC_ZIG_AP_RADIO_CHN)	MXT101 MXT151 MXT151+	apRadioChan:	0~3
0x2B – (SPC_ZIG_ACC_RELINK_INT)	MXT101 MXT151 MXT151+	edRelinkInterval:	1~255
0x2C – (SPC_ZIG_ACC_CMD_INT)	MXT101 MXT151 MXT151+	edCmdInterval:	1~255
0x2D – (SPC_ZIG_ACC_RETRY_MAX_INT)	MXT101 MXT151 MXT151+	edTimeRetryMax:	1~65535
0x2E – (SPC_ZIG_AP_REV_ALL_BC)	MXT101 MXT151 MXT151+	apRevAllBc:	0~255
0x2F – (SPC_SET_HOURMETER)	All	hourmeter:	0~71582788
0x30 – (SPC_SET_POSITION_GROUP)	All	infoGroup:	0~255
0x31 – (SPC_SET_SMART_OUTPUT2)	MXT15X MXT15X+	enable:	0 or 1
0x32 – (SPC_OUTPUT_TEST)	MXT15X MXT15X+	No Parameters	
0x33 – (SPC_OUTPUT_MACRO)	MXT15X MXT15X+	No Parameters	
0x34 – (SPC_SET_ANTI_THEFT)	MXT15X MXT15X+	enabled:	0 or 1
0x35 – (SPC_SET_BTN_PARKING)	MXT15X No CS MXT15X+No CS	enabled:	0 or 1
0x36 – (SPC_SET_DOOR_DETECT)	MXT15X No CS MXT15X+No CS	enabled:	0 or 1
0x37 – (SPC_DISARMED_MODE)	MXT15X MXT15X+	No Parameters	
0x38 – (SPC_PARKING_MODE)	MXT15X MXT15X+	enter:	0 or 1

0x39 – (SPC_ENABLE_ZIG)	MXT101 MXT151 MXT151+	enable: 0 or 1
0x3A – (SPC_CALC_ODOMETER)	All	enable: 0 or 1
0x3B – (SPC_BUZZER_CTRL)	MXT 10X MXT 10X+ MXT 100N MXT 15X	circle: 1~255 onTime: 1~255 offTime: 1~255
0x3C – (SPC_ENABLE_MICROPHONE)	MXT 10X MXT 10X+ MXT 100N MXT 15X+	enable: 0 or 1
0x3D – (SPC_MOVING_TRI_ALARM)	MXT15X MXT15X+	enable: 0 or 1
0x3E – (SPC_SET_EVENT_FLAG)	All	selectionList:16bytes events flag
0x3F – (SPC_SET_DEBOUNCE_TIMER)	All	ignDebTimer: 0~255 panicDebTimer: 0~255 ipt1DebTimer: 0~255 ipt2DebTimer: 0~255 ipt3DebTimer: 0~255 ipt4DebTimer: 0~255 maxSpeedDebTimer: 0~255 wpDebTimer: 0~255 exPowerDebTimer: 0~255
0x40 – (SPC_SET_OPT_MASK)	MXT 14X MXT15X MXT15X+	opt1Mask: 0 or 1 opt2Mask: 0 or 1 opt3Mask: 0 or 1
0x41 – (SPC_OLD_POS_TRANS_COUNT)	All	oldPosTransCount: 0~1000
0x43 – (SPC_SET_BACKDOOR_FUNC)	MXT 10X MXT 10X+ MXT 100N MXT 15X	backdoor: 0 or 1
0x44 – (SPC_SET_IGN_VOLTAGE)	MXT 14X MXT15X MXT15X+	threshold: 0~255
0x45 – (SPC_SET_OUTPUT1_INVERT)	MXT 14X MXT15X MXT15X+	enable: 0 or 1
0x46 – (SPC_OUTPUT_MACRO2)	MXT15X MXT15X+	output2ActivateTimer: 0~255 output3ActivateTimer: 0~255

0x47 – (SPC_CHARGING_ALLOW)	MXT 14X MXT15X MXT15X+	enable: 0 or 1
0x48 – (SPC_GPS_ACCELERATE_FILTER)	All	accfilter: 4~255
0x49 – (SPC_SET_SPEAKER_ENABLE)	MXT151+ MXT150_sc+ MXT120	enable: 0 or 1
0x4A – (SPC_SET_ODORPM_ENABLE)	MXT151+	odo_rpmEnable: 0~3
0x4B – (SPC_SET_RPM_THRESHOLD)	MXT151+	rpmThreshold: 0~65535
0x4C – (SPC_SET_ODOMETER_PULSES)	MXT151+	odoPulses: 0~65535
0x4D – (SPC_SET_RPM_PULSES)	MXT151+	rpmPulses: 0~65535
0x4E – (SPC_GSR_RS_MODE)	MXT15X+ MXT120	mode: 0~6
0x4F – (SPC_GSR_ROLLING_THRESHOLD)	MXT15X+ MXT120	threshold: 0~16
0x50 – (SPC_GSR_SIDE_THRESHOLD)	MXT15X+ MXT120	threshold: 0~16
0x51 – (SPC_GSR_SHOCK_THRESHOLD)	MXT15X+ MXT120	threshold: 2~16
0x52 – (SPC_INCOMING_BUZZER)	MXT10X	enable: 0 or 1
0x53 – (SPC_SMS_SEND_COUNT)	All	maxCount: 0~255
0x54 – (SPC_SMS_SENDING_INTERVAL):	All	interval: 5~65535
0x55 – (SPC_SMS_PANIC_NUMBER)	All	Index: 1~3 numSize: 0~15 number: numSize bytes
0x56 – (SPC_CFG_ALIAS_NAME)	All	aliasSize: 0~31 alias: aliasSize bytes
0x57 – (SPC_OUTPUT_MACRO3)	MXT15X MXT15X+	index: 1~3 activationDuration: 0~65535 deactivationDuration: 0~65535 cycles: 0~65535
0x58 – (SPC_KEEP_WORKING_BF_SLEEP)	All	keepWorkingTimer: 0~255
0x59 – (SPC_SET_SPEAKER_VOL)	MXT151+ MXT150_sc+ MXT120	volume: 1~10
0x5A – (SPC_SET_CALL_NUMBER)	MXT120	mode: 0 or 1 idx mode=0: 0~1 mode=1: 0~9 numSize: 0~19 number: numSize bytes
0x5B – (SPC_SET_MAKE_CALL_TIME)	MXT120	makeCallTime: 1~255

0x5C – (SPC_SET_RING_MODE)	MXT120	ringMode: 0~3
0x5D – (SPC_UPD_BY_ZIGBEE)	MXT151+	enable: 0 or 1
0x5E – (SPC_SET_ZIG_ALERT_TIME)	MXT151+	zigAlarmTime: 0~255
0x5F – (SPC_SET_ZIG_ALERT_INFO)	MXT151+	infoSize: 0~18 zigAlarmInfo: infoSize
0x60 – (SPC_SET_TIMEOUT_NOGSM)	MXT151+	timeWithoutGsm: 0~255
0x61 – (SPC_SET_TIMEOUT_MOVING)	MXT151+ MXT150_sc+ MXT120	timeoutMoving: 0~65535
0x62 – (SPC_ACTIVATE_ZIG_ALERT)	MXT151+	alertTime: 0~255
0x63 – (SPC_SET_ZIG_RSSI_FILTER)	MXT151+	zigRssiFilter: 0~15
0x64 – (SPC_SET_IGNITION_CODE)	MXT14X MXT15X+	ignCode: 0~99
0x65 – (SPC_INPUT_ON_PANIC_ANTI)	MXT15X+	bBtnDoorAsInput: 0~1
0x66 – (SPC_ENABLE_GPS_FILTER)	MXT14X MXT15X+	gpsFilter: 0~1
0x67 – (SPC_SET_AGPS_TIMER)	MXT120	timer: 0~255
0x68 – (SPC_DEV_LINK_TIMER):	MXT151+	timer: 0~255
0x69 – (SPC_NEUTRAL_SPEED_RPM):	MXT151+	neutralMaxSpeed: 0~255 neutralRpm: 0~65535
0x6A – (SPC_NEUTRAL_EXCESSIVE_RPM)	MXT151+	neutralExcessiveRpm: 0~65535
0x6B– (SPC_GPS_FAILURE_TIMER)	MXT15x+	timer: 0~255
0x6C – (SPC_SET_EXCEEDSPEED_OUTPUT)	MXT15x+	whichOutput: 0~3
0x6D – (SPC_SET_LINKFAIL_OUTPUT)	MXT151+	whichOutput: 0~3 linkFailTimes: 1~255
0x6E – (SPC_SET_JAMMING_OUTPUT)	MXT14X MXT15x+	whichOutput: 0~3
0x6F – (SPC_SET_JAMMING_ALERT)	MXT151+	enable: 0~1
0x70 – (SPC_SET_SIREN)	MXT15x+	activateDuration: 0~255 deactivateDuration: 0~255 circle: 0~255 sirenOfOpt: 1 or 2
0x71 – (SPC_SET_DISTANCE_THRESHOLD)	MXT1xx+	distanceThreshold: 0~255
0x72 – (SPC_SET_DIRECTION_THRESHOLD)	MXT1xx+	directionThreshold: 10~180
0x73 – (SPC_SET_IP_PRIORITY)	MXT1xx+	ipPriority: 0~2
0x74 – (SPC_PACKET_ENCRYPT_KEY)	MXT1xx+	packetEncryptKey: 16 bytes
0x75 – (SPC_APN2)	MXT1xx+	apn2Size: 1byte, apn2 size user2Size 1byte, user2name size pwd2Size: 1byte,

		password2 size apn2: apn2Size bytes, the apn2 user2: user2Size bytes, the user2 name password2: pwd2Size bytes, the password2
0x76 – (SPC_SET_RS_DEBOUNCE)	MXT151+	rsRollingDeb: 0~255 rsSideDeb: 0~255 rsShockDeb: 0~255
0x77 – (SPC_ANTI_ALERT_TIMER)	MXT15X+	alertTimer: 5, 10, 20
0x78 – (SPC_SET_RS_OUTPUT)	MXT15X+	rollingOpt: 0~3 sideOpt: 0~3 shockOpt: 0~3
0x79 – (SPC_SET_PARKING_OPT)	MXT15X+	parkingOpt: 0~3
0x7A – (SPC_SET_GAO_SPEED)	MXT151+	bGandOSpeed: 0 or 1
0x7B – (SPC_SET_SENDING_ORDER)	MXT1XX+ MXT120	bGrowingSending: 0 or 1
0x7C – (SPC_ENABLE_CHG_NET_SET)	iMxt	bModifyApn: 0 or 1
0x7D – (SPC_SET_ZIG_TAG_INTERVAL)	MXT1X1+ iMxt	zigTagInterval: 0~65535
0x7E – (SPC_EX_OUTPUT_CTRL)	MXT1X1+	wt200_index: 0~4 output_index: 0~2 activate: 0 or 1
0x7F – (SPC_EX_INPUT_MASK)	MXT151+	wt200 index: 0~4 mask: 0~15
0x80 – (SPC_SET_PANIC_VIA)	MXT151+	panicVia: 1~3
0x81 – (SPC_SET_TAG_FAIL_OUTPUT)	MXT151+	tagFailOpt: 0~3
0x82 – (SPC_ENABLE_AP_SET_OUTPUT)	iMxt	bApOptAllow: 0~1
0x83 – (SPC_SET_RPM_DEBOUNCE)	MXT151+	debSpeedNeutral: 0~255 debMaxExcessiveRpm: 0~255 debMaxExcessiveRpmNeutral: 0~255
0x84 – (SPC_SET_PARKING_OPT_MODE)	MXT15X+	parkingOptAct: 0~255 parkingOptDeact: 0~255 parkingOptCircle: 0~255 parkingOptInterval: 0~255
0x85 – (SPC_SET_GPS_FOR_ALERT)	MXT10X+	bGpsForAlert: 0~1
0x86 – (SPC_SET_RPM_FACTOR)	MXT151+	rpmFactor: 0~3
0x87 – (SPC_SET_GSR_DEBFACTOR)	MXT1XX+	gsrDebFactor: 0~2
0x88 – (SPC_SET_RPM_TRIGGER)	MXT151+	rpmUpTrigger: 0~1

0x89 – (SPC_SMS_SPEED_ALERT_NUM)	MXT1XX+	numSize: 0~16 number Numbersize bytes
0x8A – (SPC_RPM_EVENT_OUTPUT)	MXT151+	rpmEvt1Opt: 0~3 rpmEvt2Opt: 0~3 rpmEvt3Opt: 0~3
0x8B– (SPC_IMPROPER_MOVING_OUTPUT)	MXT14X MXT15X+	improperMovingOpt: 0~3
0x8C – (SPC_ALLOW_ANY_TAG_ACC)	MXT1X1+	anyWt110: 0~1
0x8D– (SPC_SET_LONG_TIMER_NOMOVING)	MXT15X+	ignLongTimer: 0~255
0x8E – (SPC_GSR_THRESHOLD_LEVEL)	G100	gsrLevel: 0~10
0x8F – (SPC_SET_SIM_CHIP)	MXT14X	simChip: 0~1
0x90 – (SPC_ZIG_WORKING_MODE)	MXT101+	exWorkingMode: 0~2
0x91 – (SPC_SET_IGN_VOL_FACTOR)	MXT14X	factor: 0~3
0x92 – (SPC_CALIBRATE_IGN_VOLTS)	MXT14X	No Parameter
0x93 – (SPC_SET_STOP_INTERVAL_FACTOR)	MXT14X	factor: 0~2
0x94 – (SPC_SET_INPUT1_ACTION)	MXT14X	input1Action: 0~1
0x95 – (SPC_SET_RS232_WORKING_MODE)	MXT14X	rs232WorkingMode: 0~2
0x96 – (SPC_SET_RS232_WORKING_TIMER)	MXT14X	rs232WorkingTimer: 0~255
0x97 – (SPC_SET_RS232_SPEED)	MXT14X	rs232Speed: 0~4
0x98 – (SPC_SET_RS232_RECV_TIMEOUT)	MXT14X	rs232RecvTimeout: 20~255
0x99 – (SPC_SET_RS232_ALERT)	MXT14X	rs232Alert: 0~1
0x9A – (SPC_ANTI_ALARM_AFTER_TIMER)	MXT14X	alarmAfterTimer: 0~1
0x9B – (SPC_ANTI_STATUS_INFO)	MXT14X	antiStatInfo: 0~1
0x9C – (SPC_ANTI_SILENCE_LAW)	MXT14X	antiSilence: 0~1
0x9D – (SPC_SET_RAIN_INPUT)	MXT140 MXT141	rainInput: 0~3
0x9E – (SPC_SET_RAIN_MAXSPEED)	MXT140 MXT141	rainMaxSpeed: 0~255
0x9F – (SPC_UDP_BIND_PORT)	MXT140 MXT141	bindPort: 0~1
0xA0 – (SPC_SET_ACCELERATION)	MXT140 MXT141	accelerationOutput: 0~3 accelerationLimit: 0~255
0xA1 – (SPC_SET_DECELERATION)	MXT140 MXT141	decelerationOutput: 0~3 decelerationLimit: 0~255
0xA2 – (SPC_RFID_ANY_DEVICE)	MXT14X	bAnyRfid: 0~1
0xA3 – (SPC_RFID_LOGOUT_MODE)	MXT14X	rfidLogoutMode: 0~1
0xA4 – (SPC_RFID_PASS_OPT)	MXT14X	rfidPassengerOpt: 0~1
0xA5 – (SPC_SET_TIMEZONE_EX)	MXT142	minutes: -720 ~780
0xA6 – (SPC_SET_INPUT_INVERT)	MXT141	Input: 1~3 Invert: 0~1

0xA7 – (SPC_SET_RAINSENSOR_SIGNAL)	MXT141	rainSensorSignal: 0~1
0xA8 – (SPC_SET_GENERIC_EVENT)	MXT141	input: 0~3 output: 0~2 maxspeed 0~255
0xA9 – (SPC_SET_SMART_OUTPUT1)	MXT142	smartOutput1: 0~1
0xAA – (SPC_SMS_ENABLE_ACK)	MXT142	smsAckEnabled: 0~1
0xAB – (SPC_ENGINE_SEAL_OPT)	MXT140	engineSealOpt: 0~2 engineSealCondition: 0~7 engineSealOptMode: 0~1
0xAC – (SPC_ENGINE_SEAL_RELAY)	MXT140	engineSealRelay: 0~1 engineSealDeb: 0~255
0xAD – (SPC_ENGINE_SEAL_JAMMMODE)	MXT140	engineSealJammMode: 0~1
0xAE – (SPC_ENGINE_SEAL_ACTIVATE)	MXT140	activated: 0~1
0xAF – (SPC_CAL_IGN_VOL_OUTPUT)	MXT140	calIgnVolOutput: 0~2
0xB0 – (SPC_ENGINE_SEAL_INPUT)	MXT140	engineSealInput: 0~1
0xB1 – (SPC_SENDING_INTERVAL_EX)	MXT130D	stopInterval 0, 4~65535 moveInterval 0, 4~65535
0xB2 – (SPC_ENGINE_SEAL_OPT_CYCLE)	MXT140 MXT141	EngineSealOptCycle 0, 1 EsocActMinutes 0~255 EsocDeaMinutes 0~255
0xB3 – (SPC_AGPS_ENABLE)	MXT130D	AgpsEnable: 0 or 1
0xB4 – (SPC_AGPS_SERVER)	MXT130D	UserSize: 0~63 PasswordSize: 0~31 UrlSize: 0~63 Port: 0~65535 user[]: UserSize bytes password[]: PasswordSize bytes url[]: UrlSize bytes
0xB5 – (SPC_CELL_INFO_EX)	MXT130D	lbsType: 0 ~2
0xB6 – (SPC_LBS_REQUEST)	MXT130D	None
0xB7 – (SPC_PHY_IGN_PIN)	MXT140 MXT141	phyIgnitionPin: 0~1
0xB8 – (SPC_KEEPALIVE_CLOUD)	MXT140 MXT141 MXT142 MXT130 MXT130D	keepaliveInterval: 0,3~1440
0xB9 – (SPC_GSR_FOR_IGNVOL)	MXT140 MXT141	gsrForIgnVol: 0~1
0xBA – (SPC_OPT_MODE_FOR_ACTION)	MXT140 MXT141	optModeForAction: 0~1

0xBB – (SPC_OPT_FRE_DURING_DEB)	MXT140 MXT141	optFreDuringDeb: 0~3
0xBC – (SPC_SET_ACCEL_ALARM)	MXT142	Hard Acceleration trigger: 0~8000 mG
0xBD – (SPC_SET_ACCEL_THRESHOLD)	MXT142	Debounce trigger: 0~5000 milliseconds
0xBE – (SPC_SET_DECEL_ALARM)	MXT142	Hard Braking trigger: 0~8000 mG
0xBF – (SPC_SET_DECEL_THRESHOLD)	MXT142	Debounce trigger: 0~5000 milliseconds
0xC0 – (SPC_SET_CURVE_ALARM)	MXT142	Hard Lateral trigger: 0~8000 mG
0xC1 – (SPC_SET_CURVE_THRESHOLD)	MXT142	Debounce trigger: 0~5000 milliseconds
0xC2 – (SPC_SET_CRASH_ALARM)	MXT142	Crash trigger: 0~8000 mG
0xC3 – SPC_SET_CRASH_THRESHOLD)	MXT142	Debounce trigger: 0~5000 milliseconds
0xC4 – (SPC_SET_PITCH_EVT)	MXT142	Inclination: -90 to +90 degrees
0xC5 – (SPC_SET_ROLL_EVT)	MXT142	Inclination: -90 to +90 degrees
0xC6 – (SPC_RESET_CAL_EVT)	MXT142	Reset Calibration: 0~1
0xC7 (SPC_HISTOGRAM_CONFIG)	MXT142	Number of range: 0~32 Limit of speed for each range : 0~255 Km/h
0xC8 (SPC_SET_SPEED_SUMMARY_IN_POS)	MXT142	Feature to report the speed details [0~1]: 0 - Disable ; 1 – Enable
0xC9 – (SPC_SET_OBD_ENABLED)	MXT142	Feature to enable the OBD diagnostic [0~1] 0 - Disable ; 1 - Enable
0xCA – (SPC_SET_CAN_CFG_DATASET)	MXT142	Feature to choose the dataset of each vehicle library [0~65535]
0xCB – (SPC_SET_CAN_IN_POS)	MXT142	Feature to report the CAN data in the position packet //1 byte [0~1] 0 - Disable ; 1 - Enable
0xCC – (SPC_SET_CAN_POS_CFG)	MXT142	Bit Map to specify which data that will be included in the position packet [8 bytes]
0xCD – (SPC_SET_CAN_ENABLE)	MXT142	Feature to enable CAN module (hardware) [0~1] 0 - Disable ; 1 - Enable

3.6 Power off

Inform power off event;
MT: 0x34;
DATA FIELD: 1 bytes;
UINT8 delay: // (1~255), indicates device delay in multiples of 100 milliseconds to power off.

The reply must be an ACK or NACK

Command example:
01a634189a7b0005189d04

3.7 Reset

Inform reset event.
MT: 0x35;
DATA FIELD: 1 bytes;
UINT8 delay: // (1~255) indicates device delay in multiples of 100 milliseconds to reset.

The reply must be an ACK or NACK.

Command example:
01a635189a7b0005b8d804

3.8 GPRS pause

Inform GPRS connection pause, and waits for voice call
MT: 0x36;

DATA FIELD: variable

Waiting time (1 BYTE)	Number Size (1 BYTE)	Phone number

The waiting time is in minutes (1 ~ 255)
If the Number Size is zero means allow answer any number at this time.
Otherwise only configured number can be answer.
The reply must be an ACK or NACK

3.9 GPRS resume

Device sends this command to server when the voice call has dropped or the waiting timer is expired.
MT: 0x37;
DATA FIELD: none
Server does not need to reply this command.

Command example:
01a637189a7b00ccf204

3.10 Waypoint

Add, edit, delete, modify priority and set route filter.

MT: 0x38
DATA FIELD:

ID (1 BYTE)	PARAMETER (VARIABLE)

Set ID descriptions:

0x0 - (WPO_GET_INFO): get information about waypoints
No parameter.

REPLY:

UINT8	operate	//0x0
UINT16	count	//number of waypoints stored in device
UINT16	group	//group of current searching

0x1 - (WPO_NEW): add a new waypoint to device

WAY_POINT_ITEM* pItem,		//waypoint item structure,

REPLY:
ACK or NACK

0x2 - (WPO_READ): read waypoint item

UINT16	index.	//Index of the waypoint

REPLY:

UINT8	operate	//0x2
WAY_POINT_ITEM* pItem		

0x3 - (WPO_EDIT): edit an existent waypoint
UINT16 index.
WAY_POINT_ITEM* pItem,

REPLY:
ACK or NACK

0x4 - (WPO_DEL): delete an existent waypoint
UINT16 index.

REPLY:
ACK or NACK

0x5 - (WPO_DEL_ALL): delete all waypoints
No parameter.

REPLY:
ACK or NACK

0x6 - (WPO_MOVETO): modify current waypoint's priority
UINT16 index_current
UINT16 index_moveto.

REPLY:
ACK or NACK

0x7 - (WPO_FILTER): set group want to search with.
UINT16 group.

REPLY:
ACK or NACK

The structure's definition:

```
typedef union
{
    u32 value;
    struct
    {
        u32    panic                :1    //0 do nothing, 1 set panic status
        u32    output1              :2    //0 do nothing, 1 activate, 2 deactivate
```

```
        u32     output2                 :2      //0 do nothing, 1 activate, 2 deactivate
        u32     set_forbidden_point     :1      //0 clear forbidden point
                                                //1 set forbidden point
        u32     buzzer_led_mode         :2      //0 do nothing,
                                                //1 trigger mode 1
                                                //2 trigger mode 2
                                                //3 trigger mode 3
        u32     transmission_mode       :2      //0 do nothing,
                                                //1 set GPRS transmission with ignition off
                                                //2 set GPRS transmission with ignition on
        u32     transmission_nogprs     :1      //0 do nothing
                                                //1 send one position through SMS if GPRS is not
                                                available
        u32     zig_position            :1      //0 do nothing,
                                                //1 transmit position through 2.4Ghz once
        u32     output3                 :2      //0 do nothing
                                                //1 activate,
                                                //2 deactivate
        u32     dummy1                  :2
        u32     direction:              :3      //same as position protocol
        u32     init_hour               :5      //minimum hour of day to validate point, 0 for don't care
        u32     end_hour                :5      //maximum hour of day to validate point, 0 for don't
                                                care
        u32     dummy2                  :3
    } info;
} t32_actions_validation;

typedef struct
{
    u32         id_waypoint;
    u16         wpt_group;
    u8          min_speed;              //minimum speed to validate the point - default 0x00
    u8          max_speed;              //speed above which speed excess flag will be set -
                                        default 0xFF
    t32_actions_validation      actions_conditions;
    s32         latitude_1;             //latitude of top left corner
    s32         longitude_1;            //longitude of top left corner
    s32         latitude_2;             //latitude of bottom right corner
    s32         longitude_2;            //longitude of bottom right corner
}WAY_POINT_ITEM;
```

3.11 Firmware download start

Start to download new firmware to device, after this command server must send download file frame to device. The reply after each package must be an ACK or NACK.

MT: 0x39
DATA FIELD: 5 bytes

UINT32	fileSize;	//the firmware file size.
UINT8	firmwareMode;	//the firmware mode, defined as below.

1 - UPDATE_MODE_G24HZ
2 - UPDATE_MODE_APP
4 - UPDATE_MODE_MODEM

Command example:
01a639189a7b00bcaf030002285c04
bcaf0300 – fileSize
02 – firmwareMode

3.12 Firmware download file

Frame used to send a firmware file. Each package should be limited to 1Kb.

MT: 0x3A
DATA FIELD:

OFFSET (4 BYTES)	SIZE (2 BYTES)	DATA STREAM

The reply after each package must be an ACK or NACK.

About the offset, is the position of current packet in the firmware file.
And for modem it need add 0x1000000 to different other firmware. 2.4Ghz need add 0x2000000 also. The application no need add.

Command example: (firmware file splitted into 302packets)
1st packet
01a63a189a7b000000000020034150505f ... 102400a0e1a01f00eb07f104

00000000 – offset

2003 – size

4150505f...102400a0e1a01f00eb – data stream

2nd packet

01a63a189a7b00200300002003102420a0 ... 000090e5141030a0e3a82c04

20030000 – offset

2003 – size

102420a0...000090e5141030a0e3 – data stream

Last packet

01a63a189a7b00a0ac03001c0365dbc993 ... 657074696f6e000000d05104

a0ac0300 – offset

1c03 – size

65dbc993...657074696f6e00000 – data stream

3.13 Firmware download end

Upgrades a firmware file loaded on the memory of the processor and must be sent after the Firmware File parameter. The reply must be an ACK or NACK.

MT: 0x3B
DATA FIELD: 8 bytes

UINT16	CRC_firmware;	
UINT8	firmwareMode;	//the firmware mode, defined as below.
UINT8	modemAndApp;	//if the application and modem firmware updated at same time, it must be set to 1, otherwise set to 0
UINT32	fileSize;	//the firmware file size.

1 - UPDATE_MODE_G24HZ
2 - UPDATE_MODE_APP
4 - UPDATE_MODE_MODEM

Command example:

01a63b189a7b004433021021bcaf03001ccb04

4433 – crc_firmware

02 – firmwareMode

1021 – modemAndApp

bcaf0300 – fileSize

3.14 2.4Ghz Accessory Operation

This command is used to let MXT start searching 2.4Ghz accessories or allows accessory to be joined.

MT: 0x3D
DATA FIELD:

ID (1 BYTE)	PARAMETER (VARIABLE)

Operate ID descriptions:

0x0 (AOP_SEARCH) no use now
0x1 (AOP_SEARCH_CANCEL) no use now
0x2 (AOP_SET) set accessories list

Count	Start index	End index()	ACC list
2 BYTES	2 BYTES	2 BYTES	(end – start + 1) * sizeof(zig_ed_info_t)

For the list maybe very long, the accessories list will be sent as several packets, when the server received the first packet, it should send a reply to get the other parts.

```
typedef struct
{
    u8   addr[4];
    u8   listType;
    u8   dummy[3];
} zig_ed_info_t;
```

The listType define as below:

0 Sync List
1 BroadCast List
2 Link Token List
3 Tag List

REPLY:

AOP_SET:

Count(Sum)	Start index	End index
2 BYTES	2 BYTES	2 BYTES

If the server wants to get the other parts of accessories list, it should send the AOP_SET command to request it.

ACK or NACK

if the server get all the accessories list, it should send ACK command to the device to finish the setting.

0x3 (AOP_GET) get accessories of MXT

Start index	End index
2 BYTES	2 BYTES

If the server gets the accessories list for the first time, the end index should be 0.

REPLY:

AOP_GET:

Count(Sum)	Start index	End index()	ACC list
2 BYTES	2 BYTES	2 BYTES	(end – start + 1) * sizeof(zig_ed_info_t)

0x4 (AOP_CFG_START) send accessory configuration data

Target ACC's device ID	AES key
4 BYTES	16 BYTES

REPLY:

ACK or NACK

0x5 (AOP_CFG_RESULT) send configuration result

Result	Data Length	Target DEV ID	CFG data
1 BYTE	1 BYTE	4 BYTES	0-Data length

Result: 1- configuration is ok
0 - configuration is canceled or failed
Data Length the length of CFG data
Target DEV ID: the accessory's device ID

CFG data new configuration data

REPLY:

ACK or NACK

0x6 (AOP_CFG_ORG_DATA) Send the original configuration data of the accessories

Data length	Configuration Data
4 BYTES	0-Data length

REPLY:

ACK or NACK

Definition of accessory's type:
0x20: WT100 (wireless watch)
0x21: WT110 (wireless button/tag)
0x22: WT111 (wireless temperature sensor)
0x23: WT112 (wireless switch sensor)
0x24: WT200 (wireless anti-theft/relay)
0x25: WT300 (wireless LCD)
0x26: WT400
0xFF: unknown accessories

0x8 (AOP_FREE_TEXT) send free text to accessories.

Date/Time	Text size	Message ID	Text
8 BYTES	1 BYTE	1 BYTE	Text size bytes

The date/time structure as below:

```
typedef struct
{
    u32 seconds     :6;
    u32 minutes     :6;
    u32 hours       :5;
    u32 year        :15;
    u32 month       :16;
    u32 day         :16;
} ZIG_DATE_TIME;
```

REPLY ACK after sending successfully

Command example:

01a73d1559150008699bbe0f03000d0008036d6178747261636be34304

699bbe0f03000d00 – date_Time

08 – textSize

03 – id

6d6178747261636b – text

0x9 (AOP_PRE_TEXT) send pre text to accessories.

Date/Time	Group	Message ID
8 BYTES	1 BYTE	1 BYTE

REPLY ACK after sending successfully
REPLY NACK when group is invalid, or busy.

Command example:

01a73d1559150009839cbe0f03000d000000969c04

839cbe0f03000d00 – date_Time

00 – group

00 – id

0xA (AOP_SET_PRE_TEXT) set pre text on accessories.

Group	Message ID	Text size	Text
1 BYTE	1 BYTE	1 BYTE	Text size bytes

NOTE: if Text size is 0 means delete this message.

REPLY ACK after sending successfully
REPLY NACK when group is invalid, or busy.

Command example:

01a73d155915000a001024086d6178747261636b1aa704

00 – group

1024 – id

08 – textSize

6178747261636b – text

0xB (AOP_LOAD_PRE_TEXT) load pre text from accessories.

Group	Message ID
1 BYTE	1 BYTE

REPLY the specific text. Format is:

Sub command ID	Text size	Text	Message ID
0xB	1 BYTE	Text size bytes	1 BYTE

NOTE: when request message ID is invalid, device will return next valid, and return the valid message ID.

Command example:

01a73d155915000b1021003a2b04

1021 – group

00 – id

0xC (AOP_SET_GROUP_NAME) Set group name.

Group	Name size	Name
1 BYTE	1 BYTE	Name size bytes

Command example:

01a73d155915000c000e4d6178747261636b2047726f7570d81604

00 – group

0e – nameSize

4d6178747261636b2047726f7570 – name

0xD (AOP_LOAD_GROUP_NAME) load group name.

Group
1 BYTE

REPLY the specific text. Format is:

Sub command ID	Name size	Name
0xD	1 BYTE	Name size bytes

Command example:

01a73d155915000d0003c704

00 – group

0xE (AOP_DEL_ALL_PRE_TEXT) delete all pretext text.

Group want to delete
1 BYTE

Delete group 1: 0x1
Delete group 2: 0x2
Delete group 3: 0x4
Delete group 1 and 2: 0x3
Delete group 1 and 3: 0x5
Delete group 2 and 3: 0x6
Delete group 1 and 2 and 3: 0x7

REPLY
ACK or NACK

Command example:
01a73d155915000e1021718204
1021 – group

0xF (AOP_GET_LIST_EX) Get accessories list

Type	Start index	Count
1 BYTE	2 BYTES	2 BYTES

Type : which type of list you want to get
0x0: sync list
0x1: broadcast list
0x2: link list
0x3: tag list
Start index: the offset of you want to get, start from 0.
Count: how many accessories you want to get.

REPLY:

Sub command ID	Total Count	Count	List
0xF	2 BYTES	2 BYTES	Count * 4

Total Count: the total count of the accessories which required
Count: current packet include accessories count
List: accessories ID list.

Command example:
01a73d155915000f1021000014007f5504

1021 – type

0000 – startIndex

1400 – count

0x10 (AOP_SET_LIST_EX) Set accessories list

Type	Total Count	Start index	Count	List
1 BYTE	2 BYTES	2 BYTES	2 BYTES	Count * 4

Type //which type of list you want to set
0x0: sync list
0x1: broadcast list
0x2: link list
0x3: tag list
totalCount: //the total count of accessories will be set
startIndex: //the offset of current packet will be set, start from 0.
count: //the count in current packet.
list: //accessories ID list.

REPLY
ACK or NACK

Command example:
01a73d1559150010300302000000200290a300089a7000096fa04
03 – type
0200 – totalCount
0000 – startIndex
0200 – count
290a300089a70000 – list (290a3000 – id 3148329 // 89a70000 – id 42889)

Note:
if you want erase all accessories of one type from the device, should set the “Total count” to 0.

3.15 Keep alive packet

In UDP connection mode, device will send this packet to server when keep alive timer is expired in order to update device’s IP and port. And server has no need to reply it.

MT: 0x28;
DATA FIELD: 1 byte;

UINT8 protocol;

3.16 Get old position packets

This command is used to get the old position packets on the MXT1XX.

MT: 0x3C;
DATA FIELD: mode 1 byte;

Mode is 0, get old position information
MXT1XX reply: 4bytes

Mode = 0	//1byte
Dummy	//1byte
Packets count	//2bytes

Mode is 2, get packets one by one
MXT1XX reply:

Mode = 1 or2	//1byte, 1 means current packet is the last packet.
Dummy	//1byte
Packet size	//2bytes
Packet data	

Mode is 3, cancel packets getting
MXT1XX reply:
Normal ACK

Mode is 4, get specific position count
SERVER Send:

Mode=4	
Start position count	//2bytes
Total packets	//2bytes

MXT1XX reply:
Same with mode 0's reply

e.g.
Server sends command to get 20 positions start from 100 (position count in packets). Then the command as below:

Hex data:	ASCII	
3C	position indication MT	
04	mode(info)	4
64 00	start position count	100

14 00	total count	20

After received this command device will reply as below:

Hex data:	ASCII	
3C	position indication MT	
04	mode(info)	4
00	dummy	reserved for future
14 00	total count, this count	20 maybe small than you want

Then server can send data command to get position data one by one:

Hex data:	ASCII	
3C	position indication MT	
02	mode(data)	2

Device reply:

Hex data:	ASCII	
3C	position indication MT	
02	mode(data)	2

If mode is 1 means it is the Last data packet.

00	dummy	reserved for future
36 00	the data size	54
XX XX XX XX	data, structure is same as command 31	
XX XX XX XX		

Command example: requesting 10 old positions starting from pos 31000

01a73c15591500102418790a003b0604 – command sent
1024 – mode
1879 – startPositionCount
0a00 – totalPackets

01a73c155915001024640a00940f04 – command reply
1024 – mode
64 – dummy
0a00 – totalPackets

01a73c1559150002b4d104 –position request (one by one)

01a73c1559150002644a0008ff1879e6 ... f427500077026309783c04 – position reply
(same as MT:0x31 structure)

3.17 Get ICCID

This command is used to get the ICCID

MT: 0x3E;
DATA FIELD: NONE

REPLY DATA FIELD: the string of the ICCID, variable size.

Command example:
01a73e15591500147104 – command sent

Command reply:
01a73e1559150038393535303533313331303031333030323333318ca404
38393535303533313331303031333030323333331 - ICCID

3.18 Transparent Transmission

This command is used to send the data which received from wireless accessories to the server

MT: 0x3F;
DATA FIELD: Transparent data

3.19 Transmission File Start

This command is used to transmission files to device.

MT: 0x2C
DATA FIELD: 5 bytes

File size	File type
4 BYTES	1 BYTE

The file type: 0 – AGPS Ephemeris file
The file type: 1 – CAN Library file

Reply:
Normal ACK or NACK

3.20 Transmission File Data

This command is used to transmission files to device.

MT: 0x2D
DATA FIELD:

Offset	size	File data
4 BYTES	2 BYTES	size bytes

Reply:
Normal ACK or NACK

3.21 Transmission File End

This command is used to transmission files to device.

MT: 0x2E
DATA FIELD: 2 bytes

CRC
2 BYTES

Reply:
Normal ACK or NACK

3.22 Version Packets

After power on device will send this packet to server to tell server the firmware version.
After server received this packet, also need reply same packet to device, but no need add data.

Send to server:
MT: 0x2A
DATA: 18 bytes

Modem Version	6 bytes
Application Version	6 bytes
Zigbee Version	6 bytes

In every version(6 bytes)

Version1	1 byte
Version2	1 byte

e.g. if version is 3.27 version1 = 0x3, version2 = 0x1B

Built date time 4 bytes
The structure is :

```
typedef union
    {
        u32 value;
        struct
        {
            u32 seconds     :6;
            u32 minutes     :6;
            u32 hours       :5;
            u32 day         :5;
            u32 month       :4;
            u32 year        :6;
        } info;
    } t32_built_time;
```

Reply from server
MT: 0x2A
DATA: NONE

3.23 New waypoint

MT: 0x2F
DATA FIELD:

ID (1 BYTE)	PARAMETER (VARIABLE)

Set ID descriptions:
0x0 - (WPO_GET) – Get waypoints data:
Parameter:
UINT16 index //the start index of the waypoint you want to get, from 0.

Send this command to device to get waypoint data, one time can get max 20 waypoints. Device will reply this command by ID 1 as below.

0x1 - (WPO_GET_CNF) – confirm of the get waypoint data:

Parameter:

```
UINT16          total_count:        //total waypoint stored in the device.
UINT16          select_group:       //the group used.
UINT16          index:              //the start index of the waypoint in this packet.
UINT16          count:              //the count of the waypoint in this packet.
WAY_POINT_ITEM items                //waypoint data, size is count X
                                    sizeof(WAY_POINT_ITEM) the is defined as below:

typedef union
{
    u32 value;
    struct
    {
        u32 en_panic        :2  // 0 do nothing, 1 activate, 2 deactivate
        u32 en_output1      :2  // 0 do nothing, 1 activate, 2 deactivate
        u32 en_output2      :2; // 0 do nothing, 1 activate, 2 deactivate
        u32 en_output3      :2; // 0 do nothing, 1 activate, 2 deactivate
        u32 en_max_speed    :8;
        u32 en_speed_mode   :2; // 0 no speed limit,
                                   1 use speed limit,
                                   2 use speed limit and keep it after leave
        u32 lv_panic        :2; // 0 do nothing, 1 activate, 2 deactivate
        u32 lv_output1      :2; // 0 do nothing, 1 activate, 2 deactivate
        u32 lv_output2      :2; // 0 do nothing, 1 activate, 2 deactivate
        u32 lv_output3      :2; // 0 do nothing, 1 activate, 2 deactivate
        u32 dummy           :6;
    } info;
} t32_actions;

typedef struct
{
    u32         id_waypoint;
    u16         id_group;
    u8          direction;
    u8          dummy;
    t32_actions actions;
    s32         latitude_1:         // latitude of top left corner
```

```
    s32     longitude_1:        // longitude of top left corner
    s32     latitude_2:         // latitude of bottom right corner
    s32     longitude_2:        // longitude of bottom right corner
}WAY_POINT_ITEM;
```

Remark:
Direction definition:

0	North
1	North East
2	East
3	South East
4	South
5	South West
6	West
7	North West
8 or more	any direction

Max Speed:
If the parameter “en_speed_mode”set to:

0 means the max speed limit is depended on configured in tab others. Max speed in the waypoint will be ignored

1 means the max speed limit is depended on configured in waypoint. Max speed in tab others will be ignored.

2 means the max speed limit is depended on configured in waypoint and keep it even if already leave the waypoint.

Command example:

Command sent
01a72f15591500000000e85a04
0000 – index

Command reply (3 waypoints stored on device)
01a72f155915001021030000000000300102100000010210008002a0000008597cffeac4e5efde985cffe5b675efd0300000010210008000000050000082cffe1d575bfdad53cffec28c5bfd10240000001021000800160000000d29ad0fe835d57fd6594d0fea66657fd563504
0300 – totalCount

0000 – selectGroup
0000 – index
0300 – count
1021000000 – id_waypoint – (first waypoint)
102100 – id_group
08 – direction
00 – dummy
2a000000 – actions
8597cffe – latitude_1
ac4e5efd – longitude_1
e985cffe – latitude_2
5b675efd – longitude_2
03000000 – id_waypoint – (second waypoint)
102100 – id_group
08 – direction
00 – dummy
00005000 – actions
0082cffe – latitude_1
1d575bfd – longitude_1
ad53cffe – latitude_2
c28c5bfd – longitude_2
1024000000 – id_waypoint (third waypoint)
102100 – id_group
08 – direction
00 – dummy
16000000 – actions
d29ad0fe – latitude_1
835d57fd – longitude_1
6594d0fe – latitude_2
a66657fd – longitude_2

0x2 - (WPO_SET) – set waypoint:
This command use to set all waypoint data to device, and modify the select group.
Parameter:

UINT16	total_count:	//total waypoint will be set to the device.
UINT16	select_group:	//the group used.
UINT16	index:	//the start index of the waypoint in this packet.
UINT16	count	//the count of the waypoint in this packet.
WAY_POINT_ITEM items		//waypoint data, size is count X sizeof(WAY_POINT_ITEM)

Device will reply ACK or NACK to server.

Remark:

If there are two or more waypoints have same ID, then only the first one is valid, others cannot be used by device.

Example:

If you want to append new 5 waypoints after you already have 10 waypoints, you need send command as:

total = 15, index = 10, count=5, 5 waypoints data…

If you want to save new 5 waypoints and remove old 10 waypoints, you need send command as:

total = 5, index = 0, count=5, 5 waypoints data…

If you want to delete all exist waypoints, you need send command as:

total = 0, index = 0, count=0, no waypoints data…

Command example:

01a72f155915000210210000000000102100050000000200080000000000edffffffcfffffffedffffffcfffffff2d4f04

102100 – total_count

0000 – select_group

0000 – index

102100 – count

0500000000 – id_waypoint

0200 – id_group

08 – direction

00 – dummy

00000000 – actions

edffffff – latitude_1

cfffffff – longitude_1

edffffff – latitude_2

cfffffff – longitude_2

0x3 - (WPO_MDF) – modify waypoint:
This command only modify one already exist waypoint data.
Parameter:

UINT16	index	the index of the waypoint want to modify.
WAY_POINT_ITEM	item	waypoint data

Device will reply ACK or NACK to server.

0x4 - (WPO_SEL) – modify select group:
This command only modify the select group in device.
Parameter:
UINT16 select_group, the group used.

Device will reply ACK or NACK to server.

3.24 Transparent Transmission

MT: 0x3F;
DATA FIELD:

Sub command	Parameters
1 BYTE	VARIABLE

Sub command:
0x21: Get RS232 status
No parameter.

Server sends this command to get the status of MXT14X's RS232.

Reply message please see sub command 0x23.

0x22: Set RS232 status
Parameter: Enable/disable //1 byte, 0 is disable; 1 is enable.

Server sends this command to let MXT141 enable or disable the RS232.

Reply message please see sub command 0x23.

0x23: Report RS232 status

Parameter: status //1 byte, 0 is disable; 1 is enable; 2 is forbidden to use.

This message is the ACK of the sub command 0x21 and 0x22, from device to server. It indicates current RS232 status of the device.

If device is configured rs232 disable, status will be 2

If device is configured rs232 only working when ignition on, when ignition is off, status will be 2

0x24: Data transmission from server

Parameter:

Size //2 bytes, data size
Data //Size bytes.

Server sends this command to transmit data to device.

If the RS232 is not enabled, the NACK will be sent to server.
Otherwise, data will be transferred, and reply server an ACK.

The max data size limit is 800 bytes

3.24.1 Data receive from RS232

When received the data from RS232, device will generate a new position with special protocol (0xA). And the position packet event reason is 75.

In this protocol position packet, behind all normal data, the RS232 data will attached, the structure as below:

Parameter: Size //2 bytes, data size
Data //Size bytes.

After device received data from RS232, it's waiting some milliseconds, for receive the whole packet data. When finish receiving, device will transfer them to the server by generated position.

This milliseconds can be configured, please see the "Configuration" chapter.

In one position, the max data size limit is 600 bytes, if device received packet larger than 600 bytes, it will generate more than one positions. For example, when receive 750 bytes, first position have 600 bytes, the seconds bytes have 150 bytes, these two position packets maybe have same date and

time, but position count is different. The whole packet max size must be smaller than 8192 bytes.

Server side also need reply ACK to inform device it already receive the position.

NOTE: When device reset or power on, there was a byte(0x49) will be generated and sent through RS232. This byte is not useful for transmission, only the device platform recognize byte.

3.24.2 Configurations:

RS232 Working Mode 1 byte

0: //Disable all the time

No transparent transmission feature in device.

1: //Enable only when ignition on

During ignition is turned on, RS232 can be used to transmission data.
During ignition is turned off, RS232 will working until no data received and sent timer expired. This timer can be configured, please see next parameter.

2: //Enable all the time

RS232 can work all the time, even the ignition turn off.

In mode 1 and 2, server can send command to let device open or close the RS232.

Timer to Disable RS232 1 byte 0~255 in minutes

This timer is use to close RS232 after no data transmission, it only working when mode is 1.

After ignition turn off, device will check the data receive from RS232, if no any data transmission is over the time configured, device will close RS232.

0 means no timer, RS232 will close immediately.

Baud rate 1 byte //0~4

User can select the baud rate as below:

0	9600
1	19200
2	38400
3	57600
4	115200

The default is 115200

When RS232 transmission disabled, the baud rate will change to 115200 automatically, it is for another software tool connects to device.

After transmission enable again, baud rate will change back to your configured.

Receive from RS232 timeout 1 byte, 20~255

This parameter is use to waiting the whole packet from RS232, the timeout is in millisecond, and the configured value should multiply 10.

For example, if configured to 80, it means the timeout is 800 milliseconds. The default timeout is 50(500ms).

Alert Serial 1 byte 0 or 1

If set to 1, when device generated new position, it will also send same position information through RS232, with position count as 0x0. This packet will no need any ACK.

The event 75(RS232 received event) will not alert.

3.25 Get Device Supported Parameter

This command is used to get the parameter from device, if device not support, will return NACK.

Only MXT142 and MXT130 support this command now.

MT: 0x29;
DATA FIELD: 1 byte;

UINT8 command ID;

The ID definitions please see the 3.5 chapter.
If ID is 0xFF means get all supported parameter.

Reply format is:
MT: 0x29
DATA FIELD: variable, please see the following:

ID 1byte	Supported table 32 bytes	Param1	Param2	Param3	…

ID: the command request ID;
Supported table: only when ID = 0xFF have this part.

Indicate all supported parameter at behind. This table has 32 bytes, every bit indicates one parameter, 1 means support, 0 means not support.

For example: the first byte is 0xE4, it means the parameter ID 3,6,7,8 supported in current device, and these parameters at behind field one by one.

Command example for request device parameters:
01ac29ad6cac00ff7d2104

Reply example:
01ac29ad6cac00fffeff77a044102131e4b310217810217258ae08002afa962c06000000000000000000000000000
00010211567656e65726963612e636c61726f2e636f6d2e627205636c61726f05636c61726f1567776d746b2e6
d6178747261636b2e636f6d2e62721567776d746b2e6d6178747261636b2e636f6d2e62724d164d16f0009600
3c001e00030a00050078000500102l5a001e003c00000b303331383538383235393700001021102110210000
007d7d000010210059823f40cff007390630103038008 2c08f10300ff4f0fbc7e3fffe021021000003102100102l0
0c8003700000a643c00000000000000002a00102110211021102110211464000e677072732e6f692e636f6d2e6272
00000f00000000020000000a102400000010210010210000001021207a04

01ac29ad6cac00 – SOF/DD/MT/ID
ff – command_id
feff77a044102131e4b310217810217258ae08002afa962c06000000000000000000000 – support_table
00 – dev_password
0000 – pin_puk
00 – sms_password
1021 - connect_mode
15 – apnsize
67656e65726963612e636c61726f2e636f6d2e6272 – apn
05636c61726f – user

05636c61726f – password

15 – ipsize

67776d746b2e6d6178747261636b2e636f6d2e6272 – ip1

15 – ip2size

67776d746b2e6d6178747261636b2e636f6d2e6272 – ip2

4d16 – primary_port

4d16 – second_port

f000 – keep_alive_timer

9600 – ri_stopped

3c00 – ri_moving

1e00 – ri_panic

030a00 – ri_resend

0500 – gsr_deb_moving

7800 – gsr_deb_stopped

0500 – gsr_detect

1021 – gsr_report

5a00 – gps_keep_working

1e00 – gps_unfix_timeout

3c00 – gps_coldstart_unfix

00 – sms_alias

0b3033313835383832353937 – sms_destination

0000 – sms_send_mode

1021 – led_enable

10211021 – input_enable

00 – max_speed_limit

00 – cellinfo

00 – panic_mode

7d7d – position_group

00 – antitheft

00 – btn_parking

1021 – calc_odometer

00 – moving_tri_alarm

59823f40cff007390630103038008 2c08f10300ff4f0fbc7e3fffe – event_flag

021021000003102100 – debounce_timer

102100 – output_mask

c800 – old_pos_trans_count

37 – ign_voltage

00 – output1_invert

00 – charging_allow

0a – accelerate_filter

64 – sms_sending_count

3c00 – sms_sending_interval

000000 – sms_panic_number

00 – cfg_alias_name

00 – keep_working_bf_sleep

0000 – timeout_moving

2a – ignition_code

00 – input_on_panic_panic

1021 – enable_gps_filter

1021 – gps_failure_timer

1021 – exceedspeed_output

1021 – jamming_output

14 – distance_threshold

64 – direction_threshold

00 – ip_priority

0e – apn2size

677072732e6f692e636f6d2e6272 – apn2

00 – user2size

00 – pass2size

0f – anti_alert_timer

00 – sending_order

00 – sms_speed_alert_num

00 – improper_moving_output

00 – long_timer_nomoving

02 – ign_volt_factor

00 – stop_interval_factor

00 – input1_action

00 – rs232_working_mode

0a – rs232_working_timer

1024 – rs232_speed

00 – rs232_alert

00 – anti_alarm - after_timer

00 – anti_silence_law

1021 – udp_bind_port

00 – rfid_any_device

1021 – rfid_logout_mode

0000 – timezone_ex

00 – smart_output1

1021 – sms_enable_ack

207a04 – CRC/EOF

Here is the every parameter field format:

0x1 - (SPC_DEV_PWD): device password enabled

	enabled	//1 byte, 0 means no password, 1 means have password

Command example:
01AC29AD6CAC001021AC2F04

Reply:
01AC29AD6CAC001021008D7904
00 – enable – (no password)

0x2 - (SPC_PIN_PUK): PIN and PUK

	PIN length:	//1byte
	PIN:	//PIN length bytes
	PUK length:	//1byte
	PUK:	//PUK length bytes

Command example:
01ac29ad6cac0002cf1f04

Reply:
01ac29ad6cac00020000ee3b04
00 – PIN length
00 – PUK lenght

0x3 - (SPC_DTMF_PWD): SMS password

	Enabled	//1 byte, 0 means no password, 1 means have password

Command example:
01ac29ad6cac001023ee0f04

Reply:
01ac29ad6cac000300ef1f04
00 – enabled – no password

0x4 - (SPC_CONNECT_MODE): the type of connection

UINT8	type:	//0-UDP, 1-TCP

Command example:
01ac29ad6cac001024097f04

Reply:
01ac29ad6cac0010241021599604
1021 - type

0x5 - (SPC_APN): APN user and password

UINT8	apnSize:	//apn size
UINT8*	apn:	//apnSize bytes, the apn
UINT8	userSize:	//user name size
UINT8*	user:	//userSize bytes, the user name
UINT8	pwdSize:	//password size
UINT8*	password:	//pwdSize bytes, the password

Command example:
01ac29ad6cac0005286f04

Reply:
01ac29ad6cac00051567656e65726963612e636c61726f2e636f6d2e627205636c61726f05636c61726f2acc04
15 – apnSize
67656e65726963612e636c61726f2e636f6d2e6272 – apn
05 – userSize
636c61726f – user
05 – pwdSize
636c61726f – password

0x6 - (SPC_IP_ADDR): IP and port

UINT8	primaryAddrSize:	//address size
UINT8*	primaryAddr:	//primaryAddrSize bytes, the IP address
UINT8	secondaryAddrSize:	//address size
UINT8*	secondaryAddr:	//secondaryAddrSize bytes, IP address
UINT16	primaryPort:	//the port of the IP
UINT16	secondaryPort:	//the port of the IP

Command example:
01ac29ad6cac00064b5f04

Reply:

01ac29ad6cac00061567776d746b2e6d6178747261636b2e636f6d2e62721567776d746b2e6d6178747261636b2e636f6d2e62724d164d166b5d04

15 – primaryAddrSize

67776d746b2e6d6178747261636b2e636f6d2e6272 – primaryAddr

15 – secondaryAddrSize

67776d746b2e6d6178747261636b2e636f6d2e6272 – secondaryAddr

4d16 – primaryPort

4d16 – secondaryPort

0x7 - (SPC_KEEP_ALIVE_TIMER): keep alive timer

UINT16 keepAliveTimer: //2 Bytes

Command example:

01ac29ad6cac00076a4f04

Reply:

01ac29ad6cac0007f000dfc304

f000 – keepAliveTimer

0x8 - (SPC_RI_STOPPED): report time with ignition off

UINT16 timerOfIgnitionOff;

Command example:

01ac29ad6cac000885be04

Reply:

01ac29ad6cac00089600624e04

9600 – timerOfIgnitionOff

0x9 - (SPC_RI_MOVING): report time with movement

UINT16 timerOfMovement;

Command example:

01ac29ad6cac0009a4ae04

Reply:

01ac29ad6cac00093c00e78b04

3c00 – timerOfMovement

0xA - (SPC_RI_PANIC): report time with panic
UINT16 timerOfPanic;

Command example:
01ac29ad6cac000ac79e04

Reply:
01ac29ad6cac000a1e0033b204
1e00 – timerOfPanic

0xB - (SPC_RI_RESEND) resend attempts and timeout
UINT8 attempts
UINT16 timeout

Command example:
01ac29ad6cac000be68e04

Reply:
01ac29ad6cac000b030a00d42c04
03 – attempts
0a00 – timeout

0xC - (SPC_GSR_DEB_MOVING): debounce timer when device from moving to stopped.
UINT16 timer;

Command example:
01ac29ad6cac000c1021fe04

Reply:
01ac29ad6cac000c05001adf04
0500 – timer

0xD - (SPC_GSR_DEB_STOPPED): debounce timer when device from stopped to moving
UINT16 timer;

Command example:
01ac29ad6cac000d20ee04

Reply:
01ac29ad6cac000d78002f9604
7800 – timer

0xE - (SPC_GSR_DETECT): the timer for detecting when moving
UINT16 timer;

Command example:
01ac29ad6cac000e43de04

Reply:
01ac29ad6cac000e05007ab104
0500 - timer

0xF - (SPC_GSR_REPORT): send immediately or not
UINT8 sendImmediately;

Command example:
01ac29ad6cac000f62ce04

Reply:
01ac29ad6cac000f1021a34a04
1021 – sendImmediately

0x10 - (SPC_GPS_KEEP_WORKING): GPS keep working timer
UINT16 timer;

Command example:
01ac29ad6cac001030bc2d04

Reply:
01ac29ad6cac0010305a0099f704
5a00 – timer

0x11 - (SPC_GPS_UNFIX_TIMEOUT): GPS timeout for fix
UINT16 timer;

Command example:
01ac29ad6cac0010319d3d04

Reply:
01ac29ad6cac0010311e00a1102104
1e00 - timer

0x12 - (SPC_GPS_COLDSTART_UNFIX_TIMEOUT): GPS timeout for fix when cold started
UINT16 timer;

Command example:
01ac29ad6cac0012fe0d04

Reply:
01ac29ad6cac00123c00753804
3c00 – timer

0x14 - (SPC_SMS_ALIAS): SMS alias
UINT8 aliasSize: //max 26chars
UINT8* alias: //aliasSize bytes, the alias

Command example:
01ac291e6dac001460db04

Reply:
01ac291e6dac0014086d6178747261636b629f04
08 – aliasSize
6d6178747261636b – alias

0x15 - (SPC_SMS_DESTINATION) SMS destination number
UINT8 destSize: //the size of the destination number
UINT8* destination: //destSize bytes, the destination number

Command example:
01ac291e6dac001541cb04

Reply:
01ac291e6dac00150a3331393939393838383d5b04
0a – destSize
33313939393938383838 – destination

0x16 - (SPC_SMS_SEND_MODE): set send SMS mode
UINT8 smsMode:
UINT8 allowNumberMode:

Command example:
01ac29ad6cac00167a4d04

Reply:
01ac29ad6cac001600004da404
00 – smsMode (not send)
00 – allowNumberMode (Any Number)

0x1D - (SPC_LED_ENABLE): enable led working or not
UINT8 enable;

Command example:
01ac29ad6cac001d1031fc04

Reply:
01ac29ad6cac001d1021b22f04
1021 – enable

0x1F - (SPC_INPUT_ENABLE)
UINT8 inputCount: //how many inputs on the device
UINT8 enable: //enable or disable, inputCount bytes

Command example:
01ac29ad6cac001f53dc04

Reply:
01ac29ad6cac001f10211021cc1904
1021 – inputCount

1021 – enable

0x22 - (SPC_MAX_SPEED_LIMIT): max speed limit
UINT8 maxSpeed;

Command example:
01ac29ad6cac0022ad3b04

Reply:
01ac29ad6cac002250cd7004
50 – maxSpeed

0x26 - (SPC_CELL_INFO_PRESET):
UINT8 cellInfo

Command example:
01ac29ad6cac0026297b04

Reply:
01ac29ad6cac002600fce604
00 - cellInfo

0x28 - (SPC_PANIC_MODE): trigger panic's mode
UINT8 mode;

Command example:
01ac29ad6cac0028e79a04

Reply:
01ac29ad6cac002800f3c504
00 – mode (None)

0x30 – (SPC_SET_POSITION_GROUP): information group of the position packets.
UINT8 GroupSupported
UINT8 infoGroup

Command example:

01ac29ad6cac0030de0904

Reply:
01ac29ad6cac00307d7d144104
7d – groupSupported
7d – infoGroup

0x34 – (SPC_SET_ANTI_THEFT): enable or disable anti-theft function.
UINT8 enabled

Command example:
01ac29ad6cac00345a4904

Reply:
01ac29ad6cac003400ed8304
00 – enabled

0x35 – (SPC_SET_BTN_PARKING): enable or disable local parking mode
UINT8 enabled

Command example:
01ac29ad6cac00357b5904

Reply:
01ac29ad6cac003500dcb004
00 – enabled

0x3A – (SPC_CALC_ODOMETER): enable or disable calculate odometer when ignition off.
UINT8 enable

Command example:
01ac29ad6cac003a94a804

Reply:
01ac29ad6cac003a1021c3b004
1021 - enable

0x3D – (SPC_MOVING_TRI_ALARM): activate moving trigger alarm mode.
UINT8 enable

Command example:
01ac29ad6cac003d73d804

Reply:
01ac29ad6cac003d00753904
00 - enable

0x3E – (SPC_SET_EVENT_FLAG): the event selection
UINT8 eventcount;
UINT8* supportedList, //(eventcount +7)/8 bytes
UINT8* enabledList, //(eventcount +7)/8 bytes

Command example:
01ac29ad6cac003e1030e804

Reply:
01ac29ad6cac003e59823f40cff00739063010303800 82c08f10300ff4f0fbc7e3fffe59aa04
59 – eventCount
823f40cff00739063010303800 – supportedList
82c08f10300ff4f0fbc7e3fffe - enabledList

0x3F – (SPC_SET_DEBOUNCE_TIMER): the debounce timer
ignDebTimer: //1byte
jammerDebTimer: //1byte
maxSpeedDebTimer: //1byte
wpDebTimer: //1byte
exPowerDebTimer: //1byte
input count: //1byte
iptDebTimer: //input count bytes;

Command example:
01ac29ad6cac003f31f804

Reply:
01ac29ad6cac003f021021000003102100 3b9904
02 – ignDebTimer

1021 – jammerDebTimer

00 – maxSpeedDebTimer

00 – wpDebTimer

03 – exPowerDebTimer

1021 – inputCount

00 – ipDebTimer

0x40 – (SPC_SET_OPT_MASK): output mask flags.

output count: //1byte

outputMask: //output count bytes

Command example:

01ac29ad6cac0040497704

Reply:

01ac29ad6cac0040102100127b04

1021 – outputCount

00 – outputMask

0x41 – (SPC_OLD_POS_TRANS_COUNT):

oldPosTransCount: //2bytes,

Command example:

01ac29ad6cac0041686704

Reply:

01ac29ad6cac0041c800eee004

C800 – oldPosTransCount

0x44 – (SPC_SET_IGN_VOLTAGE): voltage threshold for ignition measuring

UINT8 threshold

Command example:

01ac29ad6cac0044cd3704

Reply:

01ac29ad6cac00443700cd04

37 – threshold

0x45 – (SPC_SET_OUTPUT1_INVERT): output1 invert
UINT8 invert

Command example:
01ac29ad6cac0045ec2704

Reply:
01ac29ad6cac00450085b804
00 - invert

0x47 – (SPC_CHARGING_ALLOW): enable or disable charging when ignition off
UINT8 enabled

Command example:
01ac29ad6cac0047ae0704

Reply:
01ac29ad6cac004700e7de04
00 - enabled

0x48 – (SPC_GPS_ACCELERATE_FILTER): the accelerate filter
UINT8 accfilter

Command example:
01ac29ad6cac004841f604

Reply:
01ac29ad6cac00480a936f04
0a – accFilter

0x53 – (SPC_SMS_SEND_COUNT): max count sending through SMS when no GPRS.
UINT8 maxCount

Command example:
01ac29ad6cac00531b5504

Reply:

01ac29ad6cac005364723d04

64 – maxCount

0x54 – (SPC_SMS_SENDING_INTERVAL): the interval in seconds sending through SMS

UINT16 interval

Command example:

01ac29ad6cac0054fc2504

Reply:

01ac29ad6cac00543c00789704

3c00 – interval

0x55 – (SPC_SMS_PANIC_NUMBER): the number of sending SMS when entering panic or alerting status.

UINT8 num1Size:
UINT8 num1ber: //num1Size bytes
UINT8 num2Size:
UINT8 num2ber //num2Size bytes
UINT8 num3Size:
UINT8 num3ber: //num3Size bytes

Command example:

01ac29ad6cac0055dd3504

Reply:

01ac29ad6cac00550839383736353433320000d303331333139313931363436343634c6eb04

08 – num1Size

3938373635343332 – num1ber

00 – num2Size

0d – num3Size

303331333139313931363436343634 – num3ber

0x56 – (SPC_CFG_ALIAS_NAME): configuration alias name

UINT8 aliasSize //the size of the alias name
UINT8 alias:

Command example:
01ac29ad6cac0056be0504

Reply:
01ac29ad6cac0056084d6178747261636b1c8c04
08 – aliasSize
4d6178747261636b – alias

0x58 – (SPC_KEEP_WORKING_BF_SLEEP):
UINT8 keepworkingtimer: //1byte, in minutes

Command example:
01ac29ad6cac005870e404

Reply:
01ac29ad6cac005802e8ed04
02 – keepWorkingTimer

0x61 – (SPC_SET_TIMEOUT_MOVING): the timeout moving
UINT16 timeoutMoving

Command example:
01ac29ad6cac00610a4304

Reply:
01AC29AD6CAC00610600735304
0600 – timeoutMoving

0x64 – (SPC_SET_IGNITION_CODE): the ignition code
UINT8 ignCode

Command example:
01ac29ad6cac0064af1304

Reply:
01ac29ad6cac00642a7a0804
2a – ignCode

0x65 – (SPC_INPUT_ON_PANIC_ANTI): the key and door as input
UINT8 bBtnDoorAsInput

Command example:
01ac29ad6cac00658e0304

Reply:
01ac29ad6cac0065102142ae04
1021 – bBtnDoorAsInput

0x66 – (SPC_ENABLE_GPS_FILTER): enable or disable GPS filter
UINT8 gpsFilter

Command example:
01ac29ad6cac0066ed3304

Reply:
01ac29ad6cac006610211031fb04
1021 – gpsFilter

0x6B – (SPC_GPS_FAILURE_TIMER): GPS failure debounce timer
UINT8 timer

Command example:
01ac29ad6cac006b40e204

Reply:
01ac29ad6cac006b10214d8d04
1021 – timer

0x6C – (SPC_SET_EXCEEDSPEED_OUTPUT): output activated when exceed max speed.
UINT8 whichOutput

Command example:
01ac29ad6cac006ca79204

Reply:

01ac29ad6cac006c1021da1404
1021 - whichOutput

0x6E – (SPC_SET_JAMMING_OUTPUT): output activated when GSM jamming
UINT8 whichOutput

Command example:
01ac29ad6cac006ee5b204

Reply:
01ac29ad6cac006e1021b87204
1021 - whichOutput

0x71 – (SPC_SET_DISTANCE_THRESHOLD):
UINT8 distanceThreshold

Command example:
01ac29ad6cac00713b5104

Reply:
01ac29ad6cac007114612304
14 – distanceThreshold

0x72 – (SPC_SET_DIRECTION_THRESHOLD):
UINT8 directionThreshold

Command example:
01ac29ad6cac0072586104

Reply:
01ac29ad6cac007264a50804
64 – directionThreshold

0x73 – (SPC_SET_IP_PRIORITY):
UINT8 ipPriority

Command example:

01ac29ad6cac0073797104

Reply:
01ac29ad6cac007300b61704
00 - ipPriority

0x74 – (SPC_PACKET_ENCRYPT_KEY):
UINT8 packetEncryptKey[16]

This command is not working for MXT-142

0x75 – (SPC_APN2):

UINT8	apnSize:	//apn size
UINT8*	apn:	//apnSize bytes, the apn
UINT8	userSize:	//user name size
UINT8*	user:	//userSize bytes, the user name
UINT8	pwdSize:	//password size
UINT8*	password:	//pwdSize bytes, the password

Command example:
01ac29ad6cac0075bf103104

Reply:
01ac29ad6cac00750e677072732e6f692e636f6d2e62720000818504
0e – apnSize
677072732e6f692e636f6d2e6272 – apn
00 – userSize
00 – pwdSize

0x77 – (SPC_ANTI_ALERT_TIMER):
UINT8 alertTimer: //1byte

Command example:
01ac29ad6cac0077fd3104

Reply:
01ac29ad6cac00770f9d2a04
0f - alertTimer

0x7B – (SPC_SET_SENDING_ORDER):
UINT8 bGrowingSending: //1byte

Command example:
01ac29ad6cac007b71f004

Reply:
01ac29ad6cac007b001f9e04
00 - bGrowingSending

0x89 – (SPC_SMS_SPEED_ALERT_NUM):
UINT8 numSize: //1byte, the number size
UINT8 number[]: //numSize bytes, the number

Command example:
01ac29ad6cac00892c3f04

Reply:
01ac29ad6cac00890a33313734313835323936de102404
0a – numSize
33313734313835323936 – number

0x8B – (SPC_IMPROPER_MOVING_OUTPUT):
UINT8 improperMovingOpt: //1byte

Command example:
01ac29ad6cac008b6e1f04

Reply:
01ac29ad6cac008b00de8d04
00 - improperMovingOpt

0x8D – (SPC_SET_LONG_TIMER_NOMOVING):
UINT8 ignLongTimer: //1byte, in minutes

Command example:

01ac29ad6cac008da87f04

Reply:
01ac29ad6cac008d00782704
00 – ignLongTimer

0x91 – (SPC_SET_IGN_VOL_FACTOR):
UINT8 factor: //1byte

Command example:
01ac29ad6cac009115ac04

Reply:
01ac29ad6cac009102244104
02 – factor

0x93 – (SPC_SET_STOP_INTERVAL_FACTOR):
UINT8 factor: //1byte

Command example:
01ac29ad6cac0093578c04

Reply:
01ac29ad6cac00930010240704
00 – factor

0x94 – (SPC_SET_INPUT1_ACTION):
UINT8 input1Action: //1byte

Command example:
01ac29ad6cac0094b0fc04

Reply:
01ac29ad6cac009400939e04
00 – input1Action

0x95 – (SPC_SET_RS232_WORKING_MODE):

UINT8 rs232WorkingMode: //1byte

Command example:
01ac29ad6cac009591ec04

Reply:
01ac29ad6cac009500a2ad04
00 – rs232WorkingMode

0x96 – (SPC_SET_RS232_WORKING_TIMER):
UINT8 rs232WorkingTimer: //1byte

Command example:
01ac29ad6cac0096f2dc04

Reply:
01ac29ad6cac00960abb5904
0a – rs232WorkingTimer

0x97 – (SPC_SET_RS232_SPEED):
UINT8 rs232Speed: //1byte

Command example:
01ac29ad6cac0097d3cc04

Reply:
01ac29ad6cac00971024448b04
1024 – rs232Speed (115200)

0x99 – (SPC_SET_RS232_ALERT):
UINT8 rs232Alert: //1byte

Command example:
01ac29ad6cac00991d2d04

Reply:
01ac29ad6cac009900cfe804
00 – rs232Alert

0x9A – (SPC_ANTI_ALARM_AFTER_TIMER):
UINT8 alarmAfterTimer: //1byte

Command example:
01ac29ad6cac009a7e1d04

Reply:
01ac29ad6cac009a009cbd04
00 - alarmAfterTimer

0x9C – (SPC_ANTI_SILENCE_LAW):
UINT8 antiSilence: //1byte

Command example:
01ac29ad6cac009cb87d04

Reply:
01ac29ad6cac009c10211b0704
1021 - antiSilence

0x9F – (SPC_UDP_BIND_PORT):
UINT8 bindPort: //1byte

Command example:
01ac29ad6cac009fdb4d04

Reply:
01ac29ad6cac009f1021485204
1021 - bindPort

0xA2 – (SPC_RFID_ANY_DEVICE):
UINT8 bAnyRfid: //1byte

Command example:
01ac29ad6cac00a225aa04

Reply:
01ac29ad6cac00a21021812104
1021 – bAnyRfid

0xA3 – (SPC_RFID_LOGOUT_MODE):
UINT8 rfidLogoutMode: //1byte

Command example:
01ac29ad6cac00a31024ba04

Reply:
01ac29ad6cac00a31021b01204
1021 - rfidLogoutMode

0xA5 – (SPC_SET_TIMEZONE_EX):
SINT16 minutes: //2bytes

Command example:
01ac29ad6cac00a5c2da04

Reply:
01ac29ad6cac00a52efffb3804
2eff - minutes

0xA9 – (SPC_SET_SMART_OUTPUT1):
UINT8 smartOutput1: //1byte

Command example:
01ac29ad6cac00a94e1b04

Reply:
01ac29ad6cac00a910217bfd04
1021 – smartOutput1

0xAA – (SPC_SMS_ENABLE_ACK):
UINT8 smsAckEnabled: //1byte

Command example:
01ac29ad6cac00aa2d2b04

Reply:
01ac29ad6cac00aa102128a804
1021 – smsAckEnabled

3.26 Get Device General Information

This command is used to get general information, if device not support, will return NACK.

Only MXT142 and MXT130 support this command now.

MT: 0x50;
No parameter;

Command: **010050FFFFFFFFFD8C04**

Reply format is:
MT: 0x50
DATA FIELD: variable, please see the following:

ID 1byte	Mask 4 Bytes	Param1	Param2	Param3	…

The Mask's every bit indicate which parameter at the behind, it defind:

Bit0: Modem Version
Bit1: Application Version
Bit2: Modem Name
Bit3: Application Name
Bit4: Application Flag
Bit5: Hardware Version
Bit6: Hardware Model
Bit7: Device Password
Bit8: IMEI
Bit9: Bluetooth Address
Bit10: Encryption Key
Bit11: System Time
Bit12: SIM Card Status
Bit13: IMSI
Bit14: ICCID.

The Mask for MXT142 is 0x75EF, so it parameters defined:

Modem Version size:	//1 byte
Modem Version:	//Modem Version size bytes
Application Version size:	//1 byte
Application Version:	//Application Version size bytes
Modem Name size:	//1 byte
Modem Name:	//Modem Name size bytes
Application Name size:	//1 byte
Application Name:	//Application Name size bytes
Hardware Version:	//1 byte, now is 0xA, it means P1
Hardware Model:	//1 byte, always 1
Device Password:	//1 byte, 1 means have device password, 0 means not have
IMEI size:	//1 byte
IMEI:	//IMEI size bytes
Encryption Key:	//1 byte, always 0
SIM Card Status:	//1 byte, SIM_idle=0, SIM_plugout=1, SIM_failure=2, SIM_pin=3, SIM_puk=4, SIM_block=5, SIM_ready=6
IMSI size:	//1 byte
IMSI:	//IMSI size bytes
ICCID size:	//1 byte
ICCID:	//ICCID size bytes

Reply example:

01ac50cbd36a00ef7500001a56322e30365f4a616e2020372032303135203131 3a31313a30361a56312e35315f4a616e2020392032303135203136 3a33313a3336064d5854313432064d58543134320a1021001233353533373120303220313433383238203400060f37323430353138313231313137371438393535303533323138303033339373236363 7e2d604

50 – commandID

cbd36a00 – deviceID

ef750000 – mask (00000000000000000111010111101111)

1a – modemVersionSize

56322e30365f4a616e2020372032303135203131 3a31313a3036 – modemVersion

1a – hardwareVersionSize

56312e35315f4a616e2020392032303135203136 3a33313a3336 – hardwareVersion

06 – modemNameSize

4d5854313432 – modemName

06 – appNameSize

4d5854313432 – appName

0a – hardwareVersion

1021 – hardwareModel

00 – devicePassword

12 – imeiSize

33353533373120303220313433383238203 4 – imei (ASCII)

00 – encriptyKey

06 – SimStatus

0f – imsiSize

37323430353138313231313131373 7 – imsi (ASCII)

14 – iccidSize

38393535303533323138303033333937323636 37 – iccid

3.27 New Features Data

Below is MXT Protocol overview and explanation about the New Features Data.

MXT Protocol Package frame:

<SOF> + <DD> + <MT> + <DEVICE ID> + <DATA> + <CRC> <EOF>

DATA frame:

<PROTOCOL> + <INFO_GROUP> + <POSITION_COUNT> + <DATE_TIME> + <LATITUDE> + <LONGITUDE> + <FLAGS> + <SPEED> + <INPUT_MASK> + <INFO_GROUP_DATA> + <CELL_INFO> + <NEW_DATA>

Protocol Types:

- 0x08 – MXT Standard Package
- 0x0A – MXT Standard Package + Transparent Transmission
- 0x18 – MXT Standard Package + New Features Data
- 0x1A – MXT Standard Package + Transparent Transmission + New Features Data

New Features Data contains the information about LBS, CAN, G-Sensor and others future features.
Describe the New Features Data structure:

QTY (1 byte) + [< QTY> * < Typified Data >]
<Typified Data> = ID (1 byte) + SIZE (2 bytes) + DATA STREAM[n bytes reported in the SIZE]

Example:

2 + <ID> + <SIZE> + <STREAM> + <ID> + <SIZE> + <STREAM>

Where:
2 means the quantity of New Features Data
ID means the specific ID described in New Features Data
SIZE means the quantity of bytes that STREAM will have
STREAM means the Data described for each ID

3.27.1 New Features Data ID Types

3.27.1.1 RFU

ID = 0x01
Reserved for future use.

3.27.1.2 LBS GSM

ID = 0x02

Data structure:

<SIZE> + <RADIO_TYPE> + <LBS_INFO>

Definitions:

Size //2 bytes

Radio Type //1 byte
Where:
1 – CDMA

2 – GSM
3 – UMTS
4 – LTE

LBS Info

Age //4bytes

TimingAdvance //1 byte

Cell Towers //11 bytes for each cell tower
Where:
CellID //4 bytes
MCC //2 bytes
MNC //2 bytes
LAC //2 bytes
SignalStrengh **//**1 byte

3.27.1.3 LBS WiFi

ID = 0x03
Data structure:

<SIZE> + <WIFI_ACCESS_POINTS>

Size //2 bytes

Wifi Access Points //17 bytes
WifiNumbers //1 byte
MAC Address //8 bytes
Signal Strength //2 bytes
Age //4 bytes
Signal_to_noise_ratio_SNR //1 byte
Channel //1 byte

3.27.1.4 RFU

ID = 0x04
Reserved for future use.

3.27.1.5 Telemetry

ID = 0x05 Events

Data structure:

<SIZE> + <TELEMETRY_EVENT>+< TELEMETRY_EVENT >+…

Size // 2 bytes

Structure of Telemetry Event //10 bytes
Event Type //1 byte
Event content //9 bytes

Where:

Calibration Status
Event type: 0xC0
Content:
Status – Calibration complete //1byte [2:Accelerometer calibrated]
Reserved //8 bytes

Hard Acceleration Event (Rolling axis)
Event type: 0xC1
Content:
Maximum G-force [mili G] //2 bytes
Time to get Maximum G-force [milliseconds] // 2 bytes
Period total of event [milliseconds] // 2 bytes
Reserved //3 bytes

Hard Braking Event
Event type: 0xC2
Content:
Maximum G-force [mili G] //2 bytes
Time to get Maximum G-force [milliseconds] // 2 bytes
Period total of event [milliseconds] // 2 bytes
Reserved //3 bytes

Hard Lateral Event
Event type: 0xC3

Content:
Maximum G-force [mili G] //2 bytes
Time to get Maximum G-force [milliseconds] // 2 bytes
Period total of event [milliseconds] // 2 bytes
Lateral side [0: Left side ; 1: Right side] // 1 byte
Reserved //2 bytes

Impact Detect Event
Event type: 0xC4
Content:
Maximum G-force [mili G] //2 bytes
Time to get Maximum G-force [milliseconds] // 2 bytes
Period total of event [milliseconds] // 2 bytes
Reserved //3 bytes

3.27.1.6 Telemetry

ID = 0x06 Delta of Journey

Data structure:

<SIZE> + <DELTA OF JOURNEY>

Size // 2 bytes

We have some groups separated by related features and all them are identified with specific ID. For each group be included in the Delta of Journey is necessary that at least one feature of this group will be read in device side.

Example: If device not read at least RPM features, the group 0xA0 will not be included in the Delta of Journey automatically.

Structure of Delta of Journey:

Version of Delta [number of version] //1byte
Reason of generate the Delta [Always 0xFF, means that ignition control the trigger] //1 byte

ID [0xA0] means RPM items // 1 byte

Size of group [number of group's byte] // 1 byte
RPM range blue [number of seconds] // 3 bytes
RPM range green [number of seconds] // 3 bytes
RPM range yellow [number of seconds] // 3 bytes
RPM range red [number of seconds] // 3 bytes
RPM maximum [number] //2 bytes
RPM mean [number] //2 bytes

ID [0xA1] means detail of speed //1byte
Size of group [number of group's byte] //1 byte
Maximum speed during the journey [km/h] //1 byte
Mean speed during the journey [km/h] //1 byte
Hodometer accumulated during the journey [meters] //3 bytes

ID [0xA2] means journey's timer
Size of group [number of group's byte] //1 byte
Time of journey [seconds] //3 bytes
Time moving during journey [seconds] //3 bytes
Time stopped during journey [seconds] //3 bytes
Time with vehicle engine working [seconds] //3 bytes

ID [0xA3] means reference and location items
Size of group [number of group's byte] //1 byte
Data/time when finish the journey // 4 bytes
Position count of start journey //2 bytes
Position count of final journey //2 bytes
Latitude and Longitude of start journey //8 bytes
Latitude and Longitude of final journey //8 bytes

ID [0xA4] Accelerometer items
Size [number of group's byte] //1 byte
Maximum acceleration (3 axis) [Rolling axis front (2 bytes), Rolling axis back (2 bytes), Lateral axis absolute (2 bytes)] //6 bytes
Average acceleration (3 axis) [Mean Rolling axis front (2 bytes), Mean Rolling axis back (2 bytes) , Mean Lateral axis absolute (2 bytes)] //6 bytes

ID [0xA5] RFU

.
.
.

3.27.1.7 Telemetry

ID = 0x07 Position's Information

Data structure:

<SIZE> + <POSITION'S INFORMATION>

Size // 2 bytes

Structure of Position's Information (The data are provided by CAN):

Bit Map to specify which data that will be included in the position packet. The digital inputs listed below always will be send and this configuration only allows the firmware's analysis //8 bytes

Rpm	//1 bit;
Speed	//1 bit;
Odm	//1 bit;
Clutch	//1 bit;
Brake	//1 bit;
ParkingBrake	//1 bit;
MotorBrake	//1 bit;
FuelLevel1	//1 bit;
FuelLevel2	//1 bit;
EngineTemp	//1 bit;
FuelConsumption	//1 bit;
WsWipers	//1 bit;
DoorClosed	//1 bit;
DoorLocked	//1 bit;
SeatBelt	//1 bit;
Headlights	//1 bit;
Trunk	//1 bit;
IntakeAirTemp	//1 bit;
IntakeAirFlow	//1 bit;
ThrottlePosition	//1 bit;
BarometricPressure	//1 bit;
ControlModuleVoltage	//1 bit;
AirTemperature	//1 bit;
FuelType	//1 bit;
EthanolRatio	//1 bit;
OilTemperature	//1 bit;

EngineFuelRate	//1 bit;
EngineRefTorque	//1 bit;
MalfunctionIndLamp	//1 bit;
CurrentGear	//1 bit;
DTCs	//1 bit;
dummy1	//33 bits;

Digital inputs data. The values listed below are activated only when the relative bit is set to “1”. When the relative bit is set to “0” means that input is deactivated //4 bytes

Clutch	//1 bit;
Brake	//1 bit;
ParkingBrake	//1 bit;
MotorBrake	//1 bit;
WsWipers	//1 bit;
DoorClosed	//1 bit;
DoorLocked	//1 bit;
Trunk	//1 bit;
SeatBelt	//1 bit;
Headlights	//1 bit;
MalfunctionIndLamp	//1 bit;
dummy	//22 bits;

Analogic data. It depends of Bit Map’s configuration and it will be sent on order, as shown below: //32 bytes

Speed	//2 bytes;
Rpm	//2 bytes;
Odometer	//4 bytes;
FuelLevel1	//1 byte;
FuelLevel2	//1 byte;
FuelConsumption	//2 bytes;
IntakeAirTemp	//2 bytes;
IntakeAirFlow	//2 bytes;
ThrottlePosition	//1 byte;
BarometricPressure	//1 byte;
ControlModuleVoltage	//1 byte;
AirTemperature	//1 byte;
FuelType	//1 byte;
EthanolRatio	//1 byte;

OilTemperature	//1 byte;
EngineTemperature	//1 byte;
EngineRefTorque	//3 bytes;
CurrentGear	//1 byte;
EngineFuelRate	//3 bytes;
NumDTCAvailable	//1 byte;
NumDTCPacket	//1 byte;
DTC Packet	// (NumDTCPacket * 2) bytes;

If NumDTCAvailable is greater than zero, each packet will send (last data on structure) with predefined length of DTC (10 DTCs) and this value will be decremented where the packets remainder will be next in next position;

3.27.1.8 Histogram of Speed

ID = 0x08 Histogram of Speed

Data structure:

<SIZE> + <HISTOGRAM OF SPEED>

Size // 2 bytes

This feature generates a histogram of speed reported by device. The package of histogram should be generated every o'clock (12:00:00, 13:00:00, 14:00:00, 15:00:00...) when the speed is different of zero Km/h.

Structure of Histogram of Speed:

Last active package timestamp //4 bytes - same structure of t32_date_time
Actual active package timestamp //4 bytes - same structure of t32_date_time
Limit of speed to each range [value from 0 to 255 km/h] //1 byte
Number of range [value from 0 to 32; 0 disable the Histrogram feature] //1 byte
(Number of range + 1) * 2 bytes // n bytes

Last range always is about time where device not get GPS FIX. When the speed is calculated by CAN this last range (GPS NOT FIX) always will have value 0.

3.27.1.9 Event Reconstruction (until 30 seconds)

ID = 0x09 Event Reconstruction

Data structure:

<SIZE> + <EVENT RECONSTRUCTION>

Size // 2 bytes

This data will be sent when occur any Telemetry Event configured by user.

Structure of Event Reconstruction:

Time to generate the event's historic (0~ 30 seconds; 0: disable the feature) //1 byte
Bit map (1 means enable ; 0 means disable) //1 byte

1//: Rolling axis front [mili G] //2 bytes
1//: Rolling axis back [mili G] //2 bytes
1//: Lateral axis right [mili G] //2 bytes
1//: Lateral axis left [mili G] //2 bytes
1//: Lateral axis absolute [mili G] //2 bytes
1//: Vertical axis positive [mili G] //2 bytes
1//: Vertical axis negative [mili G] //2 bytes
1//: Speed value and RPM [1byte: speed [km/h] ; 1byte RPM [value*100] //2 bytes

(Number of seconds configured)*(Number of bits enabled) *2 bytes //n bytes

3.27.1.10 RFU

ID = 0x10
Reserved for future use.

3.27.1.11 RFU

ID = 0x11
Reserved for future use.

3.27.1.12 Route's reconstruction

ID = 0x12 Route's reconstruction

<SIZE> + <ROUTE'S RECONSTRUCTION>

Size // 2 bytes

Structure of Route's Reconstruction:

Initial Latitude [GG.DDDDDD * 1000000] //4 bytes
Initial Longitude [GGG.DDDDDD * 1000000] //4 bytes
Initial Speed [Km/h] //1 byte
Number of Sections (Section means seconds incremented in route's reconstruction) // 1 byte
(Number of Sections)*3 bytes //n bytes

Where these 3 bytes are:

Latitude variation [only decimal part = [(-127 ~ +128) **/** (100000)] //1 byte
Longitude variation [only decimal part = [(-127 ~ +128) **/** (100000) //1 byte
Speed [km/h] // 1 byte

Example for reconstruct the latitude and longitude of each sector:

-	Latitude	Longitude	Speed
Initial Value	-10.000001	-10.000001	5 km/h
First variation	(-50 /100000)	(-60/100000)	17 km/h
First variation calculed	-10.000501	-10.000601	17 km/h
Second variation	(+30/100000)	(-40/100000)	23 km/h
Second variation calculed	-10.000201	-10.001001	23 km/h

3.27.1.13 RFU

ID = 0x13

Reserved for future use.