

BOX

tracker

Communication Protocol v1.5



BOX telematics limited
4 Roman Park
Roman Way
Coleshill
West Midlands
B46 1HG

T: +44 (0)1675 434200
F: +44 (0)1675 434300

Legal and notice information.

© Copyright 2001–2009 BOX telematics ltd.

BOX telematics makes no warranty of any kind with regard to this material, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose. BOX telematics shall not be liable for errors contained herein or for incidental or consequential damages in connection with the furnishing, performance, or use of this material.

This document contains proprietary information, which is protected by copyright. No part of this document may be photocopied, reproduced, or translated into another language without the prior written consent of BOX telematics limited. The information is provided “as is” without warranty of any kind and is subject to change without notice. Nothing herein should be construed as constituting an additional warranty. BOX telematics shall not be liable for technical or editorial errors or omissions contained herein.

Preface

This document describes the server communication and configuration protocols used by BOX-tracker. It provides technical information to allow developers to integrate its functionality into a server application for vehicle tracking or asset management.

The protocol is based on a TCP connection over the Internet via GSM GPRS and supports both connect and drop or a permanent server connection. The protocol is based on 7bit ASCII comma separated values, making it easy to develop and provides features for sending logged data, configuration settings and firmware updates.



Warning:

This document supersedes all previous provisional release documents prior to 1 February 2009. Provisional commands and features not detailed in this documentation are no not currently supported and may stop working in future releases of firmware.

Technical Support

For help and assistance please email TechnicalSupport@box-tracker.com.

Document Conventions

Data sent from the tracker to the server is shown in **red** and data sent from the server to the tracker is shown in **blue**. An  symbol also indicates data being sent from the server and  indicates data being sent to the tracker, these symbols do not form part of the data being sent, they are documentation markers to help clarify the direction of communication. Optional parameters or fields that may be left blank are shown enclosed in square brackets.

Contents

1	Document History.....	5
2	Communication Methods	6
2.1	GPRS (TCP) Connection	6
2.2	Commands	7
3	Commands sent by BOX-tracker.....	8
3.1	Header Command (H)	8
	Log Command (L)	10
3.1.1	Data Items	11
4	Firmware Update Protocol	15
4.1	Firmware Available.....	15
	⇒ FA	15
4.2	No Firmware Available.....	15
	⇒ N	15
4.3	Firmware Transfer	15
	F[,FirmwareVersion,StartPosition]	15
	F,FirmwareVersion,StartPosition,Length	15
5	Configuration Update Protocol	16
6	Configuration Commands	17
6.1	⇒ ACCESS.....	17
6.2	⇒ ADDKEY.....	17
6.3	⇒ ANC.....	18
6.4	⇒ ANL.....	18
6.5	⇒ CONFIG.....	18
6.6	⇒ DEFAULT.....	19
6.7	⇒ DELKEY.....	19
6.8	⇒ DIR.....	19
6.9	⇒ FWS_TIME.....	19
6.10	⇒ GEOF#.....	20
6.11	⇒ GPRS_APN.....	20
6.12	⇒ GPRS_USERNAME.....	20
6.13	⇒ GPRS_PASSWORD.....	20
6.14	⇒ GPSHC.....	21
6.15	⇒ GPSD.....	21
6.16	⇒ GPSLI.....	21
6.17	⇒ GPSLI_BOFF.....	21
6.18	⇒ GPSLI_IOFF.....	22
6.19	⇒ IMMOBILISER.....	22

6.20	⇒ IMMOB	22
6.21	⇒ IN	23
6.22	⇒ IDLEALMT	23
6.23	⇒ IGN_OFF_CHG	23
6.24	⇒ JDISTALM	23
6.25	⇒ JTIMEALM	24
6.26	⇒ LOG	24
6.27	⇒ ODOMETER	24
6.28	⇒ ODOMETER_TRIP	25
6.29	⇒ OFFSW	25
6.30	⇒ OUT#	25
6.31	⇒ PANIC_BTN	25
6.32	⇒ PASSWORD	26
6.33	⇒ PERM_CON	26
6.34	⇒ RESET	26
6.35	⇒ SAVE	26
6.36	⇒ SERVER1_ADDR	27
6.37	⇒ SERVER1_PASS	27
6.38	⇒ SERVER1_PORT	27
6.39	⇒ SERVER2_ADDR	27
6.40	⇒ SERVER2_PASS	28
6.41	⇒ SERVER2_PORT	28
6.42	⇒ SIM_SN_ALWAYS	28
6.43	⇒ SPEEDALM	28
6.44	⇒ SYNC	29
6.45	⇒ TIME	29
6.46	⇒ UPLOADI	29
6.47	⇒ UPLOADI_BOFF	29
6.48	⇒ UPLOADI_IOFF	29
6.49	⇒ UPLOADD	30
6.50	⇒ UPLOADHC	30
6.51	⇒ VALIDATE_KEY	30
6.52	⇒ VBAT12	30
6.53	⇒ VER	31
7	Appendix	32
7.1	CRC Check Sum Algorithm	32
7.2	Example Communication Sessions	33
7.2.1	Disconnected Session Sequence	33
7.2.2	Permanent connection	33
7.2.3	Firmware Download	34
7.2.4	New configuration	34
7.2.5	Configuration update	35
7.3	Road Map	36

1 Document History

Ver	Date	Section	Description
1.00	1/11/2008		Preliminary release
1.01	2/2/2009		Various update prior to formal release
1.03	16/4/2009	3.2.1	Event codes 35 & 36 added
		5.3	Battery logging details added
		5.6	FWS_TIME now defaults to 3600seconds
		5.26	PERM_CON (permanent connection) added
		5.32	SERVER2_ADDR added
		5.33	SERVER2_PASS added
		5.34	SERVER2_PORT added
		5.35	SIM_SN_ALWAYS added
		5.42	VBAT12 added
		6.3	Road map updated
1.5	19/6/09	6.15	GPSD added
		6.49	UPLOADD added
		6.50	UPLOADHC added
		3.2.1	Event code 24 modified and event code 25,39,40,41 added.
		6.10	GEOFx added
		6.19	IMMOBILISER added
		6.51	VALIDATE_KEY added
		6.2	ADDKEY added
		6.7	DELKEY added
		6.20	IMMOB added
		7.3	Road map updated
		5	Configuration Update Protocol
		7.2.4	Added New configuration example session

2 Communication Methods

Commands can be sent to and from BOX-tracker via several methods depending on setup:

- Remote connection over GPRS (TCP)
- Local RS232 connection
- Local USB connection
- Remote connection over GSM (CSD)

To configure via its serial port requires a VT100 terminal emulation program such as HyperTerminal, configured to 115200 baud, 8 data bits, no parity, 1 Stop bit, no flow control.

Commands sent by SMS text message to the tracker must take the following format:

>password,command1,value;command2,value;command3,value.....

Where password is the password of the tracker – see password command.

2.1 GPRS (TCP) Connection

TCP over GPRS is the main communication method and all connections are initiated by BOX-tracker connecting to the server using a TCP socket. Connections initiated from the server to BOX-tracker are not supported. If required an SMS command can be sent to BOX-tracker to request it to initiate a connection with the server otherwise it will connect based on configurable settings.

Once a TCP connection has been made BOX-tracker will pass all logged information without the server having to make a request. In this situation the communication between BOX-tracker and the server can be minimal making it easy and quick to develop a communication interface. During a connection the server may also send commands to BOX-tracker thus providing two way communications. If a connection is interrupted BOX-tracker will initiate a new connection and resume communication without loss of data as long as these protocols are followed.

When a connection is first opened a header command is sent to the server. This command identifies the tracker and provides the server the ability to verify if the connection is valid for security reasons. The header does not need any response from the server and can be ignored if preferred.

Following the header the tracker will send log commands containing the data logged since the last connection and which have not been sent to the server successfully.

After sending the logged commands BOX-tracker will send an END (E) command to notify the server all data has been passed. The server should respond with an ACK (A) command so the tracker knows the data has been received, thus preventing it from being sent again.

BOX-tracker will process any command the server has sent prior to the above ACK command. If no commands have been sent or once they have been processed the tracker will drop the connection, unless configured to keep it open. If the connection is configured to remain open then logged data will be sent in real-time at intervals based on configurable settings.

The following shows a typical communication session:

```
⇐ H,BT,358281002435893,081028142432,F5813D19,6D6E6DC2
⇐ L,081028142429,G,52.51084,-1.70849,0,170,0,1,0
⇐ L,081028142432,G,52.51081,-1.70849,0,203,0,16,0
⇐ E,1
⇒ A,1
```

2.2 Commands

All communication is 7bit ASCII based.

A command consists of an initial word, often abbreviated to a single character, followed by one or more optional parameters. Neither the command name nor the parameters are case sensitive, unless otherwise documented. Any leading or trailing spaces within the command name or each parameter are ignored. Each command sent must be terminated by a Carriage Return (0xD) or within an SMS by a percentage (%) symbol. The only exception is a command sent via SMS where the last command need not be terminated.

A command may contain mandatory or optional parameters. Parameters are separated by a comma and if a parameter is not required it can be omitted. If a single parameter contains a space, comma or semicolon then it must be enclosed in double quotation marks (0x34). Quotation marks may not be contained within a string parameter.

Note: Legacy commands or features not documented here should not be used as they may stop working in future versions of BOX-tracker.

3 Commands sent by BOX-tracker

The following are the primary commands used during a GPRS connection to pass logged information from the tracker to the server.

3.1 Header Command (H)

⇐ **H, DeviceType,SerialNum,DateTime,SecurityToken,FirmwareVer[,SimNumber]**

When a TCP connection is first made BOX-tracker will send the header command. The header command is never sent via SMS, instead a simple password is included.

If the header command is not received within 15 seconds of a connection first being made, or instead any other command is received, it is suggested that the server drop the connection as an unknown source may be attempting to connect to the server.

The header command consists of the single H character followed by several parameters.

⇐ **H,DeviceType,SerialNum,DateTime,SecurityToken,FirmwareVer[,SimNumber]**

eg

⇐ **H,BT,N878123,080415081234,D63E6DD9,6D6E6DC2,8944100300825505377**

Device Type	= BT
Serial Number	= N78123
Date & Time	= 2008-04-15 08:12:34
Security Token	= D63E6DD9
Firmware	= 6D6E6DC2
SIM number	= 8944100300825505377

Device Type The device Type defines the type of device the server is communicating with. Currently BOX-tracker only returns the string **BT**. In the future other Device Type strings may be returned to identify different version or types of device, allowing the server to handle communication differently according to protocols required.

The device type identifier also allows this protocol to be used by other tracking suppliers who choose to implement it. They would provide a different Device Type and this would identify their tracker from BOX-tracker.

For most cases this identifier can be ignored.

Serial Number The serial number is the number which uniquely identifies each BOX-tracker. The serial number can be found on the BOX-tracker case.

Date Time The current date and time of the trackers real-time clock. Whilst BOX-tracker normally synchronises its time from the GPS receiver this value can be used by the server to verify the time is correct and send a command to correct it. However Its main purpose is to provide a constantly changing security token.

The date and time is formatted as **YYMMDDHHmms**.
e.g. **090201033229** would be 1st February 2009 3:32:29am

Security Token The Security Token allows the server to validate and authenticate an incoming connection. The number contained in the security token is based on defined rules such that it changes each time but can be verified as being valid. If so desired the server can ignore this number but by ensuring the security token is valid (i.e. adheres to the algorithm detailed below) there is a higher probability the connection is from a known source.

The Security Token is based on a standard CRC32 (Cyclic Redundancy Checksum) algorithm (see appendix for an example) and passed as 8 hexadecimal digits.

The Security Token is calculated by concatenating the Serial Number, Date Time string (as passed in the header command), and a secret password (which can be configured) into a single string and using this to obtaining a CRC32 value. Commas are not included in this string. The default secret password is "password", without the quotation marks. This password can be changed. The CRC calculation is based on a case sensitive string.

For a header of

↳ **H,BT,358281000146500,080401153209,9F84265E**

The security token is calculated on the string

35828100014650008041153209password

Where

358281000146500	= is the Serial Number
080401153209	= is the DateTime (2008-04-01 15:32:09)
password	= your hidden password which has a default of "password"
9F84265E	= 8 digit security token

The security token is passed as an 8 digit Hexadecimal formatted number.

Firmware This field contains the current firmware CRC as 8 hexadecimal digits. See firmware update for further details.

SIM Number The SIM number of the card currently inserted is returned in the header command. However as this number changes infrequently it is not passed in every header command, instead it is included once every 48 hours or the first time after the SIM has been changed. This helps reduce communication overheads. If present the server may use this to update local records.

If required the SIM number can be configured (via the ⇨ SIM_SN_ALWAYS command) to always be present. This allows the SIM number to be used to identify the tracker instead of the serial number.

Log Command (L)

```
⇐ L, <DateTime>,<DataItem>,<Values>;<DataItem>,<Values>;<DataItem>,<Values>
```

The Log command is used to pass historical (or real-time) log entries recorded by BOX-tracker. The log command consists of a single **L** character followed by the date and time, Data Item and its optional parameters.

A single log command can contain multiple Data Items where each is separated by a semi-colon (;) character (0x3B). A Data Item is only ever included once in a Log Command but each Data Item (listed below) is not necessarily included in each log command nor can the order of the Data Items be assumed to be consistent.

It is therefore important to parse the log command by splitting on semi-colons and checking the name of each Data Item before handling the values associated with the Data Item.

Date Time The date and time is always in the format **YYMMDDHHmmss** and is the time the log record was made. If multiple Data Items are included in the log command then all were recorded at this time.

Data Item BOX-tracker can log a variety of Data Items, the name of which is passed prior to its values. The Data Item name allows the server to identify how to handle its values. The following Data Item names are currently used; additional Data Items may be used in the future. (See later for a detailed description of each)

Name	Description
G	GPS position with event code, speed, direction and distance
END	End of journey information
DID	Driver ID as a string
GF	Geofence entry or exit
DIO	Digital Input 1 with a value of 0 or 1
DI1	Digital Input 2 with a value of 0 or 1
AI0	Analogue Input 1 with it integer value
AI1	Analogue Input 2 with it integer value
AI2	Analogue Input 3 with it integer value
AI3	Analogue Input 4 with it integer value
BOOT	Reboot reason
FAULT	Fault reason as a string

Values The values passed after a Data Item name depends on the Data Item. Some may only have a single value others may have many, each separated by a comma. See the following for specific details of what is included.

3.1.1 Data Items

↳ **G, Latitude, Longitude, Speed, bearing, Distance, EventCode, Status**

The main Data Item is **G**, which is used to pass all GPS location data. Seven values are passed after the G Data Item name each separated by a comma:

- Latitude** Latitude is a signed floating point number representing degrees and fractions of a degree between -90.0 and +90.0. North is positive, South is negative.
- Longitude** Longitude is as a signed floating point number representing degrees and fractions of a degree between -180.0 and +180.0. East is positive, West is negative.
- Speed** A decimal number specifying the speed the vehicle was travelling (in kph) when the log was made.
- Bearing** A decimal number specifying the direction of travel when the log was made. The bearing is specified in whole degrees (0 to 360).
- Distance** By default this contains the distance travelled since the start of the journey (travel). This can be configured to return the odometer reading if required. Distance is returned as a floating point number in Kilometers.
- Event Code** The event code defines the reason why the log entry was made. The following event codes are returned, however, future firmware releases may return additional event codes. The event code is a decimal number.

Value	Description
1	Interval timer. The BOX-tracker can be configured to log locations at configurable intervals. This event indicates the log occurred as a result of the interval timer.
2	GPS heading has changed. This only occurs whilst the ignition is on.
3	Poll Request Via SMS. A command has been received via SMS requesting the location to be logged and passed to the server.
4	Tracker pushbutton pressed. When the BOX-tracker pushbutton is pressed a connection is made to the server.
5	External Panic Button pressed. The external panic button has been pressed.
6	Ignition on.
7	Ignition off.
18	GSM signal jamming has been detected.

19	GPS Tamper. Occurs when the GPS antenna is removed or a fault occurs.
20	Journey Distance Exceeded. This event is logged when the distance of the current journey exceeds a configurable value.
21	Journey Time Exceeded This event is logged when the length of the current journey exceeds a configurable time.
22	Speeding The speed of the vehicle has exceeded a configurable value. The speed of the vehicle must drop by 20% before this event will occur again.
24	Geofence Entered A geofence has been entered
25	Geofence Exited A geofence has been exited
28	External Power On (See the VBAT12 command) The external power (vehicle battery) has been turned on. Usually in a vehicle this will only occur once when first installing or when the vehicles battery is reconnected after servicing.
29	External Power off (See the VBAT12 command) The external power (vehicle battery) has been removed.
32	Idle Event Occurs when the tracker remains stationary with the ignition on for more than a configurable period of time.
34	Internal battery low
35	External battery low (See the VBAT12 command)
36	External battery failed (See the VBAT12 command)
39	Allow start with driver ID key (See the IMMOB command)
40	Prevent start (See the IMMOB command)
41	Allow start without driver id key (See the IMMOB command)

Status This defines the status of the vehicle at the time the log was made. The value is passed as a hexadecimal number each bit within this number has a specific meaning as described below.

Bit	Value = 0	Value = 1
0	Ignition is off	Ignition is on.
1	Vehicle is stationary	Vehicle is moving.
2	True position	Last known position ^{#1}

^{#1} When a GPS fix is unavailable the last know position is returned, Bit 2 indicates if this has occurred.

eg. **L,080328112435,G,52.4,-1.3,101,94,23.89,1,1**

Time stamp = 2008-03-23 11:24:35
 Latitude = 52.4 degrees north (north is positive and south is negative)
 Longitude = 1.3 degrees west (east is positive and west is negative)
 Speed = 101 kph
 Bearing = 94 degrees (0 degrees = north)
 Distance = 23.89 kilometers since ignition on (may be blank in certain applications)
 Event Code = 1
 Status = 1 (ignition on)

↳ **End, MaxSpeed, IdleTime, StartTime**

Returns information relating to the last journey. This Data Item is passed when an ignition off occurs.

MaxSpeed Maximum speed which occurred during the journey, specified as a decimal number in Kph

IdleTime Total number of seconds the vehicle was idle during the journey. (Decimal number)

StartTime The time the journey started in the format YMMDDHHmmss

eg. **L,080328112435,END,68,450,080328104720**

↳ **DID, value**

Returns the driver ID number (as a string) of the driver key presented by the driver.

eg. **L,080328112435,DID,0xC9A3ED000000** Driver key C9A3ED000000.

↳ GF#, value

Returns a value which indicates if a Geofence was entered or exited

Defines the geofence number affected. A decimal number (1 to 64)

Value 1 = Entry, 0 = Exit

eg.

L,090617113820,G,52.51045,-1.70872,2,22,0.13,24,3;GF2,1 Geofence 2 entered.

L,090617123820,G,52.54045,-1.72872,2,22,0.13,25,3;GF2,0 Geofence 2 exited

↳ DI#, value

Defines the status of a specific digital input.

DI# The Data Item name where # defines the digital input number (0 or 1)

Value Either the value 0 or 1, indicating the state of the input

↳ AI#, value

Defines the status of a specific Analogue input.

AI# The Data Item name where # defines the analogue input number (0 to 3)

Value Floating point number representing the measured Voltage

↳ BOOT, message

Message is a string and explains the reason why BOX-tracker has been rebooted.

↳ FAULT, message

Message is a string which describes a fault that has occurred on BOX-tracker.

4 Firmware Update Protocol

Firmware updates are controlled by the server notifying BOX-tracker that a new version is available. BOX-tracker then requests the server to send the firmware. The firmware is sent as a stream of bytes over the same TCP connection. Should the connection drop during this process then on reconnecting BOX-tracker will request the server for the remaining bytes not received. This method allows for reliable updates, even when GSM coverage is unreliable, without requesting the same data multiple times.

At no time should the server start sending the firmware without having received a request from BOX-tracker to do so.

If preferred, firmware updates may be collected from the servers hosted at BOX telematics by setting the FWS_TIME parameter (See section 6.9).

4.1 Firmware Available

⇒ FA

The server should check the firmware version number passed in the Header (H) command to see if a later version is available. If a new version is available then the server should send an FA command to the tracker.

On receipt of the FA command, and after all logs have been sent, BOX-tracker will send an F Command (see below) requesting the server to start stream the firmware data.

4.2 No Firmware Available

⇒ N

Whilst there is no reason for this to happen, if BOX-tracker were to request a version of firmware which does not exist then this command can be sent by the server to terminate the upload request.

4.3 Firmware Transfer

⇐ F[,FirmwareVersion,StartPosition]
⇒ F,FirmwareVersion,StartPosition,Length

FirmwareVersion The firmware version is obtained from the filename of the firmware file, without the extension. Currently this is also the CRC32 check sum of the data contained within the file but this may change in the future.

StartPosition	The start position requested by BOX-tracker in the F command or zero if a file different to that being requested is being sent. This is formatted as a decimal number.
Length	The length of the firmware data (from the first byte to the last). This is formatted as a decimal number.

On receipt of an **FA** command BOX-tracker will send an **F** command, without any parameters, to request the server to start sending the latest firmware. If firmware data is available the server should send an **F Command** in response to this request. Immediately following this command the server should start sending the firmware data as a stream of bytes.

Should the download be interrupted BOX-tracker will resume the download by sending an **F Command** with the firmware version partly received and the last position within the firmware data that it received successfully. The server should respond by sending an **F Command** containing the firmware version, start position (as requested) and the remaining number of bytes being sent. It should then start sending the firmware data from the position requested.

If on resuming a download the server has a new firmware version available then instead of resuming the sending of the requested firmware it may instead send the new firmware from the start. If this happens the **F Command** should be sent with the new firmware version, a start position of zero and the length of the firmware data. BOX-tracker will see the firmware version is different to that requested and will start receiving the new firmware from the start.

5 Configuration Update Protocol

If on connecting the tracker has no configuration (because it has never been programmed or because it has been cleared (defaulted)), it will send a **CN Command** to request all configuration setting. The server will stream all the commands configured for the unit and at the end the server will send an **E command** and await receipt of the **A command**.

If on connection to the server the server identifies it has new commands to send to the tracker it will send a **CR command** and await the tracker to request them. On receipt of **CP command** the server will stream just the pending commands (those changed since the last time the tracker requested them). At the end the server will send an **E command** and await receipt of the **A command**. On receipt of the "A" command the tracker will record the date the commands were last sent so they are not sent again.

6 Configuration Commands

The following commands may be sent to the tracker to configure it to behave in different ways or to requests a specific action.

6.1 ⇒ ACCESS

This commands allows the password access requirement to be turned on or off for specific communication methods. A single decimal number follows; each bit within this number relates to a communication method and defines if a password is required before access is allowed.

If a bit is true (1) a password is required and access won't be allowed until this is entered. If the bit is zero then no password is required when first communicating with the tracker, via the specified method.

This command has no affect in controlling communication over GPRS. GPRS communication is initiated by the tracker and uses a more comprehensive security method.

Bit	Access Method
0	RS232 Command line
1	USB Command line
2	GSM Command Line
3	SMS Command Line

E.g. USB and SMS requires a password

ACCESS,10

Default value: 12 (password required for GSM and SMS access).

Range: Decimal number representing a single 8-bit byte

See also the ⇒ PASSWORD command (section 6.32)

6.2 ⇒ ADDKEY

This command is used in conjunction with the immobilisation & Validate keys feature.

Addkey adds a new key number to the driver ID key table held in non-volatile memory. The key identifier length cannot exceed 18 characters. The key string must contain only alpha numeric characters i.e. a-z, A-Z and 0-9.

E.g. add key 123456789

addkey,1,123456789

When a driver ID key is presented (by Dallas iButton® or other electronic key reader) the key table is searched and if the key is found the vehicle is allowed to start via the immobilisation feature.

If the key is not found in the table the buzzer will beep (if enabled and fitted).

See also IMMOBILISER , VALIDATE_KEY, IMMOB, DELKEY

6.3 ⇒ ANC

Enabling or disabling logging of analogue channels 0 to 3 and the externally battery supply. Each parameter may be set to a 0 (disable logging) or 1 (enable logging)

- Parameter 1 = Analogue Channel 0
- Parameter 2 = Analogue Channel 1
- Parameter 3 = Analogue Channel 2
- Parameter 4 = Analogue Channel 3
- Parameter 5 = External battery

To enable logging analogue 0 and analogue 1
ANC,1,1,0,0,0

To enable logging the external battery supply
ANC,0,0,0,0,1

Default value: 0 for all parameters
 Range: Each parameter can be 0 or 1

6.4 ⇒ ANL

Defines the rate (in seconds) the enabled analogue channels are logged (See ANC command). If this value is set to zero then no Analogue channels will be logged even when enabled via the ANC command.

Default value: 0 seconds
 Range: 0 to 4,294,967 (47 days)

6.5 ⇒ CONFIG

Returns the current settings, the order and specific settings returned may vary depending on the firmware version.

⇒ CONFIG

```
ACCESS      12
ANC         0,0,0,0,0
ANL         0
FWS_TIME    3600
GPRS_APN    ""
GPRS_PASSWORD "pass"
GPRS_USERNAME "user"
GPSD        0
GPSHC       0
GPSLI       60
GPSLI_BOFF  86400
GPSLI_IOFF  86400
IDLEALMT    0
IGN_OFF_CHG 0
IMMOBILISER 2
```

```
IMMOB      0
JDISTALM   0
JTIMEALM   0
ODOMETER_TRIP 1
OFFSW      1
OUT0       0
OUT1       0
PANIC_BTN  1
PASSWORD   "password"
SERVER1_ADDR ""
SERVER1_PASS "password"
SERVER1_PORT 0
SERVER2_ADDR ""
SERVER2_PASS "password"
SERVER2_PORT 0
SIM_SN_ALWAYS 0
```

```
SPEEDALM 0
UPLOADD 320
UPLOADHC 1
UPLOADI 300
UPLOADI_BOFF 86400
```

```
UPLOADI_IOFF 86400
VALIDATE_KEY 1
VBAT12 1
OK
```

6.6 ⇒ **DEFAULT**

The default command sets all settings to its default state (factory default). Most alarms and events are disabled.

6.7 ⇒ **DELKEY**

This command is used remove a driver ID key from the key table that has previously been added using the addkey command.

E.g. to delete key 123456789ABC from group A enter
`delkey,A,123456789ABC`

The ⇒ **DELKEY,ALL** command will delete all keys (in all groups).

6.8 ⇒ **DIR**

Returns a list of all files on the tracker.

6.9 ⇒ **FWS_TIME**

If the firmware update protocol (section 4) is not being implemented on your system then firmware updates can be obtained from the servers at BOX Telematics.

This parameter defines how often the tracker checks these servers for updates. If this is set to 0 then no checks will be made and any firmware will need to be updated in accordance with the update protocol in section 4.

Default value: 3600 (Do not check BOX Telematics for updates)

Range: 0 to 2,147,483,646 seconds

6.10 ⇒ **GEOF#**

Defines the size and location of Geofence zones that when breached generate logged events 24 & 25

GEOF<index>,1,alarm_entry,<alarm_exit>,<lat>,<long>,<rad>

<index>	64 geofences can be created this value index can between 1 and 64
<alarm_entry>	(1 Enabled, 0 disabled) If enabled then event is recorded when tracker enters geofence
<alarm_exit>	(1 Enabled, 0 disabled) If enabled then event is recorded when tracker exits geofence
<lat>	Geofence centre latitude
<long>	Geofence centre longitude
<rad>	Radius of geofence in km as a decimal number

E.g. to configure geofence 2:

geof2,1,1,0,53.58688,-2.40346,2.79

6.11 ⇒ **GPRS_APN**

Allows a specific GPRS APN to be specified.

If this is set to an empty string then the public APN contained in the trackers internal APN table is used, based on the network of the SIM. For a private APN this will always need to be set.

E.g. **GPRS_APN,OrangeInternet**

Default value: "" A public APN is automatically detected.

Range: 0 to 64 characters

6.12 ⇒ **GPRS_USERNAME**

Allows a specific GPRS APN user name to be specified. If **GPRS_APN** is set to an empty string then this parameter is ignored.

Default value: user

Range: 0 to 30 characters

6.13 ⇒ **GPRS_PASSWORD**

Allows a specific GPRS APN password to be specified. If **GPRS_APN** is set to an empty string then this parameter is ignored.

Default value: pass

Range: 0 to 30 characters

6.14 [⇒ GPSHC](#)

This causes a log to be made each time the vehicles direction changes by more than the specified number of degrees. If this is set to zero then no log is made on a heading change.

To configure the tracker to log an entry when the direction changes by 45 degrees or over

GPSHC,45

Default value: 0 degrees (disabled)

Range: 0 to 360 degrees

6.15 [⇒ GPSD](#)

When configured each time the vehicles distance travelled is greater than this parameter value GPS data is logged. The distance travelled measurement is reset each time any log event occurs. If this parameter is set to zero then no log is made on distance travelled.

Eg: Configure the tracker to log an entry when the distance travelled is greater than or equal 5km

GPSD,5

Default value: 0 km (disabled)

6.16 [⇒ GPSLI](#)

Defines the interval (in seconds) between successive vehicle location logs whilst the ignition is on. This only defines the interval between logs and not the frequency the data is sent to the server.

Eg: Log vehicle locations every two minutes

GPSLI,120

Default value: 60

Range: 0 to 4,294,967 seconds (49.7 days)

6.17 [⇒ GPSLI_BOFF](#)

Defines the interval (in seconds) between successive vehicle location logs whilst the external power supply is off. This only defines the interval between logs and not the frequency the data is sent to the server.

Eg: Log locations every hour whilst the external power supply is off

GPSLI_BOFF,3600

Default value: 86400 seconds (24 hours)

Range: 0 to 4,294,967 seconds (49.7 days)

6.18 ⇒ **GPSLI_IOFF**

Defines the interval (in seconds) between successive vehicle location logs whilst the ignition is off. This only defines the interval between logs and not the frequency the data is sent to the server.

Eg: Log locations every hour whilst the ignition is off

GPSLI_IOFF,3600

Default value: 86400 seconds (24 hours)
Range: 0 to 4,294,967 seconds (49.7 days)

6.19 ⇒ **IMMOBILISER**

This functionality allows the BOX-tracker unit to control one or both of its outputs to provide the immobilisation feature. The immobilisation feature enables and disables the vehicles ignition cranking circuit depending on the state of digital output 1 and uses digital output 0 to sound a buzzer.

IMMOBILISER,<0-2>

- 0 = Immobiliser is not fitted and the out commands are enabled
- 1 = Immobiliser relay fitted. Command out1 is disabled
- 2 = Immobiliser relay and buzzer fitted (out commands are disabled)

Default value: 0

See Also IMMOB, VALIDATE_KEY, ADDKEY, DELKEY, OUT#

6.20 ⇒ **IMMOB**

The IMMOB command specifies how the immobiliser operates in relation to driver ID key reading and also allows for remote immobilisation via SMS, GSM & GPRS. When the IMMOB command a GPS event code is recorded.

IMMOB,< Value >

Value	Description	Logged GPS event code
0	Allow the vehicle to start if a valid Ibutton is presented	39
1	Prevent the vehicle from starting	40
2	Allow the vehicle to start	41

Default value: 0

See Also IMMOBILISER, VALIDATE_KEY, ADDKEY, DELKEY, OUT#

6.21 ⇒ IN

Requests the current status of all inputs, it is mainly used for diagnostic support.

```
⇒ IN
↳ Digital: 0 0
↳ Analogs: 0027 0027 0025 0027 0945
↳ Analogs: 0.105V 0.105V 0.1V 0.105V 22.942V
↳ Ignition: 1
↳ OK
```

6.22 ⇒ IDLEALMT

Specify the time (in seconds) for which the tracker must be stationary with the ignition on before an idle event code 32 is logged.

This setting only applies to vehicle trackers. A value of zero disables logging completely

To set the tracker to log idle time greater than 20 seconds

```
IDLEALMT,20
```

Default value: 0 (disabled)
Range: 0 to 4,294,967 seconds (49.7 days)

6.23 ⇒ IGN_OFF_CHG

When set to zero the internal battery is not recharged when the ignition is off. When set to a 1 the internal battery is recharged from the external supply when the ignition is off. When the ignition is on the internal battery is always recharged from the external supply (car battery)

```
IGN_OFF_CHG,1
```

Default value: 0 (Do not recharge with ignition off)
Range: 0 or 1

6.24 ⇒ JDISTALM

If a journey exceeds the number of kilometres specified by this command then event code 20 is logged. Set to zero to disable

To log an event when journeys exceed 200 km

```
JDISTALM,200
```

Default value: 0
Range: Kilometres, as a positive floating point number, or zero

6.25 ⇒ **JTIMEALM**

After the ignition has been turned on for the specified the time event code 21 is logged. Set to zero to disable

To log an event when journeys exceed 2 hours

JTIMEALM,7200

Default value: 0

Range: 0 to 4,294,967 seconds (49.7 days)

6.26 ⇒ **LOG**

The log command can be used to log (with timestamp) user specific data to the tracker. This data will be sent to the server on the next connection. The server must understand what to do with this data so the format of the data needs to be agreed this with the server provider.

The data must

- not have any non ASCII characters
- be enclosed in double quote characters if it contains spaces
- not contain double quote characters in the data
- not exceed 70 characters in length

E.g. these two examples produce exactly the same result

log temp=12.34,humidity=45.67,io=0x45

log "temp=12.34,humidity=45.67,io=0x45"

E.g. some text with spaces

log "This is some user specific data"

6.27 ⇒ **ODOMETER**

Used to set or query the odometer reading.

To query the current odometer reading

ODOMETER,?

To set the odometer to a specific value reading

ODOMETER,10345

Default: 0

Range: Kilometres, as a positive floating point number or zero

6.28 ⇒ **ODOMETER_TRIP**

Each log entry contains the distance travelled since the start of the journey. If this is set to 1 then the distance is reset to zero at the end of each journey and the distance acts as a trip meter. If it is set to zero the distance is not reset and the distance acts as an odometer reading.

Default value: 1 (Enable trip readings - reset odometer at end of each journey)
Range: 0 or 1

6.29 ⇒ **OFFSW**

Enable or disable the use of the trackers inbuilt push button switch to turn the tracker off. Set to 1 to allow the tracker to be turned off and 0 to prevent this.

To prevent the off pushbutton turning the tracker off
OFFSW,0

Default value: 1 (enabled)
Range: 0 or 1

6.30 ⇒ **OUT#**

Allows the trackers digital output to be turned on or off. The output number should be used in place of the # in the command name

Set output 0 on (high)
OUT0,1

Set output 1 off (low)
OUT1,0

Default: 0
Range: 0 or 1

6.31 ⇒ **PANIC_BTN**

If enabled (set to 1) and digital input 0 becomes high a panic event code 5 is logged and sent to the server.

Default value: 0
Range: 0 or 1

6.32 ⇒ **PASSWORD**

This defines the password which is used to calculate the security token that needs to be entered when using:

- RS232 command line
- USB command line
- GSM Command line
- Sending an SMS message to be processed by the tracker

The password is used to calculate a security token and it is this security token which should be used to gain access via one of these communication channels, not the password. The security token is calculated by appending this password to the trackers serial number and calculating a CRC32 bit checksum (see 7.1). The following would give a security token of "4596CB7F".

358281000146500password

This password is "not" used to calculate the security token passed in the header command sent over GPRS, instead see Server1_Pass command (section 6.37). The password is case sensitive.

E.g. **PASSWORD,myPassword**

Default value: "password"
Range: 0 to 20 characters

6.33 ⇒ **PERM_CON**

Enables or disables permanent TCP connections with the server. A permanent connection can help reduce the amount of GPRS data used and reduce costs.

E.g. To enable permanent connections
PERM_CON,1

Default value: 0
Range: 0 or 1

6.34 ⇒ **RESET**

Restart the trackers processor, restoring the last saved settings.

6.35 ⇒ **SAVE**

Configuration settings are held in trackers volatile memory and need to be saved to allow them to be restored when the tracker is reset. Commands sent over GPRS are automatically saved when the connection is dropped, however, this command may be sent to ensure they are save immediately.

6.36 ⇒ **SERVER1_ADDR**

The TCP/IP address used for all primary communication over GPRS. The address may be specified as an IP address or as domain name URL. If set to an empty string then no connection is made over GPRS.

E.g. **SERVER1_ADDR,myserver.tracker.com**
SERVER1_ADDR,62.231.42.78

Default value: "" (no connection)
Range: 0 to 64 characters

6.37 ⇒ **SERVER1_PASS**

This password used to calculate the security token contained within the Header Command sent by the tracker when first connecting to the server. This password is case sensitive.

E.g. **SERVER1_PASS,MyPassword**

Default value: password
Range: 0 to 20 characters

6.38 ⇒ **SERVER1_PORT**

The port number to be used for GPRS communication. Setting to zero will disable all GPRS communication.

E.g. **SERVER1_PORT,9002**

Default value: 0
Range: 0 to 65535

6.39 ⇒ **SERVER2_ADDR**

A secondary fall back TCP/IP server address may be specified. If the tracker fails to connect to the SERVER1 address it will attempt to do so via the SERVER2 if this is configured.

The address may be specified as an IP address or as domain name URL. If set to an empty string then no attempt will be made to connect to an alternative server.

E.g. **SERVER2_ADDR,myserver2.tracker.com**
SERVER2_ADDR,62.231.42.79

Default value: "" (no connection)
Range: 0 to 64 characters

6.40 ⇒ **SERVER2_PASS**

This password used to calculate the security token contained within the Header Command sent by the tracker when first connecting to the server. This password is case sensitive. (See SEVER2_ADDR)

E.g. **SERVER2_PASS,MyPassword**

Default value: password
Range: 0 to 20 characters

6.41 ⇒ **SERVER2_PORT**

The port number to be used for GPRS communication. Setting to zero will disable all GPRS communication. (See SEVER2_ADDR)

E.g. **SERVER2_PORT,9002**

Default value: 0
Range: 0 to 65535

6.42 ⇒ **SIM_SN_ALWAYS**

Normally the SIM number is only passed in the TCP Communication header command when the SIM is first changed or once every 48 hours. This is intended to reduce the amount of data passed when making a connection.

If this command is set to 1 then the SIM number is passed in each header command allowing the tracker to be identified by its SIM number instead of its serial number.

E.g. **SIM_SN_ALWAYS,1**

Default value: 0
Range: 0 or 1

6.43 ⇒ **SPEEDALM**

Specify a speed that if exceeded will cause an event code 22 to be logged. Once this event is recorded the speed must drop by 20% of the alarm speed before another speed event is recorded.

Speeds are specified in Kilometres per Hour and a value of zero disables this feature.

E.g. **SPEEDALM,80**

Default value: 0
Range: Kilometres, as a positive floating point number or zero.

6.44 ⇒ SYNC

Requests the tracker to make a connection with the server over GPRS. This command is particularly useful when sent via SMS. It has no parameters.

6.45 ⇒ TIME

Sets the trackers real-time clock. The time is automatically set when a GPS fix is achieved but this command may also be used.

To set the trackers time to 31st March 2009 at 15:13:01

TIME,20090131151301

6.46 ⇒ UPLOADI

On Asset trackers this defines the frequency data is sent to the server and is the only parameter which has any control over this. On Vehicle trackers this defines the frequency data is sent to the server whilst the ignition is ON.

E.g. **UPLOADI,60**

Default value: 300 seconds (5 minutes)
Range: 0 to 4,294,967 seconds (49.7 days)

6.47 ⇒ UPLOADI_BOFF

On Vehicle trackers this defines the frequency data is sent to the server whilst the external battery supply is OFF. On Asset trackers this has no affect.

E.g. **UPLOADI_BOFF,10800**

Default value: 86400 seconds (24 hours)
Range: 0 to 4,294,967 seconds (49.7 days)

6.48 ⇒ UPLOADI_IOFF

On Vehicle trackers this defines the frequency data is sent to the server whilst the ignition is OFF. On Asset trackers this has no affect.

E.g. **UPLOADI_IOFF,7200**

Default value: 86400 seconds (24 hours)
Range: 0 to 4,294,967 seconds (49.7 days)

6.49 ⇒ **UPLOADD**

On Vehicle trackers this defines distance in km between each upload event. The accrued distance travelled is reset each time any upload event occurs. If this parameter is set to zero then upload is not initiated on distance travelled.

E.g. **UPLOADD,7**

Default value: 0 (off)

6.50 ⇒ **UPLOADHC**

On Vehicle trackers this causes the unit to upload on the angle entered using the command GPSHC.

E.g. **UPLOADD,7200**

Default value: 0 (Do not upload on heading change)

Range: 0 & 1

6.51 ⇒ **VALIDATE_KEY**

This feature is used in conjunction with the immobiliser & addkey functions

0 = Do not validate key. (default)

Any Ibutton Key presented to the reader is accepted as valid and the immobiliser is disabled allowing the vehicle to start.

1 = validate key

Any Ibutton Key presented to the reader is looked up in the stored "keys" file. If the key is not present in the file then the key is rejected (buzzer sounds for 1 second if fitted & enabled) and the immobiliser is not disabled.

E.g. All keys must be validated before immobiliser is deactivated

VALIDATE_KEY,1

See Also IMMOBILISER, IMMOB, ADDKEY, DELKEY,

6.52 ⇒ **VBAT12**

To accurately detect when the external battery is low the tracker needs to know the normally external supply voltage. Setting to 1 indicates 12 volts and 0 indicates 24 volts.

This command does "not" affect or configure the trackers operating voltage range (which is 9 to 36 volts), it only affects the voltage level where event codes 35 (Power Failed), 36 (Power Low) ,28 (Power On) and 29 (Power Off) are logged. This threshold being lower for 12 volt supplies than 24 volt supplies.

E.g. To define a 24 volt supply
VBAT12,0

Default value: 1 (12 Volt)
Range: 0 or 1

6.53 ⇒ VER

Returns version and firmware information on the tracker

⇒ **VER**

↳ **1.12**

↳ **OK**

⇒ **VER 1**

↳ **\$URL: file:///C:/SvnRepos/Tracker/Tags/1.12/version.c \$: Aug 1 2008 12:58:41**

↳ **GCC: 4.1.1**

↳ **FW: id = 5AA5A55A, len = 68492, crc = D23795E8**

↳ **OK**

7 Appendix

7.1 CRC Check Sum Algorithm

The following code may be used to calculate CRC Security Token to include in the Header command (3.1) and to access the tracker via GSM, USB or SMS. This routine can return either a 16 or 32 BIT CRC. BOX-tracker uses a 32 Bit CRC.

The “data” parameter in the following CrcChecksum routine is the byte array conversion of the string value which consists of the remote device Serial Number, Data & Time (as passed in the header command) and hidden password.

The CRC32 checksum uses a polynomial of 0x04C11DB7 (or 0xEDB88320 when bit reflected)

```
public enum CrcType {
    CRC16CCITT = 0,
    CRC32     = 1
}

private static readonly int[] crcBit    = { 16, 32 };
private static readonly long[] crcPoly  = { 0x1021, 0xEDB88320 };
private static readonly bool[] crcReflectIn = { false, true};
private static readonly bool[] crcReflectOut = { false, true };
private static readonly bool[] crcXor    = { false, true };

public static long CrcChecksum(byte[] data, CrcType crcType) {

    if (data.Length == 0) return 0;

    int crclIndex = (int)crcType;
    int bits      = crcBit[crclIndex];
    long poly     = crcPoly[crclIndex];
    bool reflectIn = crcReflectIn[crclIndex];
    bool reflectOut = crcReflectOut[crclIndex];
    bool xorOut    = crcXor[crclIndex];
    long outMask   = ((long)1 << bits) - 1;
    long checksum  = outMask;

    for (int i = 0; i < data.Length; i++) {

        byte bt    = data[i];
        long maskIn = (reflectIn) ? 0x1 : 0x80;
        long maskOut = (reflectOut) ? 0x1 : 0x1 << (bits-1);

        for (int n = 0; n <= 7; n++) {

            if (((checksum & maskOut) != 0) ^ ((bt & maskIn) != 0)) {
                if (reflectOut) checksum >>= 1; else checksum <<= 1;
                checksum ^= poly;
            }
        }
    }
}
```

```

else {
    if (reflectOut) checksum >>= 1; else checksum <<= 1;
}

if (reflectIn) maskIn <<= 1; else maskIn >>= 1;
}
}

if (xorOut) checksum = ~checksum;
checksum &= outMask;

return checksum;
}

```

7.2 Example Communication Sessions

7.2.1 Disconnected Session Sequence

```

⇐ H,BT,358281000146500,080401153209,9F84265E,12F4A98D
⇐ L,080528112835,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528112935,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113035,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113135,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113235,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ E,0
⇒ A,0

```

(Connection Dropped)

```

⇐ H,BT,358281000146500,080401153209,9F84265E,12F4A98D
⇐ L,080528113335,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113435,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113535,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113635,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113735,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ E,0
⇒ A,0

```

7.2.2 Permanent connection

```

⇐ H,BT,358281000146500,080401153209,9F84265E,12F4A98D
⇐ L,080528112835,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528112935,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113035,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113135,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113235,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113335,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113435,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113535,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113635,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113735,G,52.4,-1.3,101.6,94,23.89,1,03

```

⇐ L,080528113835,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528113935,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ E,0
⇒ A,0
⇐ L,080528115135,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528115235,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528115335,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528115435,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528115535,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ E,1
⇒ A,1
⇐ L,080528115635,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528115735,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528115835,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528115935,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ L,080528125135,G,52.4,-1.3,101.6,94,23.89,1,03
⇐ E,2
⇒ A,2

7.2.3 Firmware Download

⇐ H,BT,358281000146500,080401153209,9F84265E,12F4A98D
⇒ FA
⇐ L,080528112501,AI1,145.56
⇐ L,080528112630,DI3,1
⇐ L,080528112730,FAULT,A very bad fault has occurred
⇐ L,080528112835,G,52.4,-1.3,101.6,94,23.89,1,03;DI3,1
⇐ L,080528112935,DEBUG,Failed connections = 23
⇐ L,080528152501,AI1,145.56
⇐ L,080528152601,AI1,145.56
⇐ L,080528152701,AI1,145.56
⇐ L,080528152701500,AI1,145.56
⇐ E,0
⇒ A,0
⇐ F
⇒ F,49F66CE3,0,123975
⇒
⇒ 123975 bytes of file data
⇒

7.2.4 New configuration

⇐ H,BT,358281005562966,090615124343,CE4DD65C,27002FD3
⇐ L,090615124145,AI4,10.67
⇐ L,080528112835,G,52.4,-1.3,101.6,94,23.89,1,03;DI3,1
⇐ E,1
⇒ A,1
⇐ CN
⇒ IGN_OFF_CHG,1
⇒ OFFSW,0

⇒ PANIC_BTN,1
⇒ VBAT12,1
⇒ ANC,0,0,0,0,1
⇒ E,2
⇐ A,2

7.2.5 Configuration update

⇐ H,BT,358281000146500,080401153209,9F84265E,12F4A98D
⇒ CR
⇐ L,080528112501,AI1,145.56
⇐ L,080528112630,DI3,1
⇐ L,080528112835,G,52.4,-1.3,101.6,94,23.89,1,03;DI3,1
⇐ L,080528152501,AI1,145.56
⇐ L,080528152601,AI1,145.56
⇐ L,080528152701,AI1,145.56
⇐ L,080528152701500,AI1,145.56
⇐ E,0
⇒ A,0
⇐ CP
⇒ JDISTALM,200
⇒ JTIMEALM,7200
⇒ UPLOADI,60
⇒ E,1
⇐ A,1

7.3 Road Map

- Harsh braking, harsh acceleration detection.
- Driver ID not presented within time limit log Event 14.
- More status information is to be passed with each GPS position.
- Configuring analogue inputs as digital inputs (to give maximum of 6 Digital inputs).
- Digital status passed back as a HEX value with a GPS position with each packet (configurable).
- Instant GPRS connection on a digital status change.
- Accelerometer used to detect motion or an Impact.
- Binary UDP protocol.
- Peripheral hardware connectivity
- CANBUS interface