# GPS VEHICLE TRACKER
# WFP-300B

## SERVER DATA
## COMMUNICATION PROTOCOL

Version - 1.1

## *1.1 LOGIN PACKET:*

Whenever the device power up/system reset, first it will send the login packet to know the device protocol, model and firmware version to the server.

Data Format:
$WTLOGIN, IMEI, DEVICE MODEL,MODULE, HW:VER, SW:VER, Geozone dbVer,Operator,00#

Sample Data:
$WTLOGIN,P01,869988016656635,WFP-300B, STD200,HW:2.0,SW:2.1h, DB:,IND AIRTEL,00#

| Field | Data | Description |
|---|---|---|
| $ | | Each packet starts with $ |
| WTLOGIN | | Message ID |
| Protocol Ver | P01 | Device Protocol Version |
| IMEI | 869988016656635 | Device IMEI Number 15Digit |
| DEVICE MODEL | WFP-300B | Device model number |
| RELEASE | STD200 | Custom Release Version |
| HW:VER | HW:1.0 | Hardware version |
| SW:VER | SW:2.1h | Firmware Version |
| DB: | GeoZone db | Geozone database file Version |
| Operator | IND AIRTEL | GSM operator |
| CheckSum | 00 | Checksum of data packet from $ to # |
| # | | Packet End with # |

## *1.2 GPS PACKET:*

Data Format:
$WTGPS, Protocol Ver, IMEI, DateTime, Livedata, GPS Status, Latitude, Longitude, Altitude, Speed, Direction, Odometer, Moving Status, ExtBatteryVolt, IntBatteryVolt, GSMSignal , LAC, Cell id, MCC, MNC, AlertMsgCode,SensorInterface,IGN, Analog Input1, Digital Input1, Output1, CheckSum #

Sample Data:

$WTGPS,P01,869988016656635,20140101070500,1,1,13.0965700,80.2913900,10.4,0.36,128.43,2.500,1,12.6, 4.2|1,25,63,3EAF,40,404, AA|0,0,1,2.5, 1,1,000,4A#

| Field | Data | Description |
|---|---|---|
| $ | | Packet starts with $ |
| WTGPS | | Message ID |
| Protocol Ver | P01 | Device Protocol Version |
| IMEI | 869988016656635 | Device IMEI Number 15Digit |
| Date Time | 20140101070500 | Date Time in format YYYYMMDDHHMMSS |
| Live data | 1 | 1 –Online Live data, 0 – Offline Log Data |
| GPS Status | 1 | GPS Data Packet 1–Valid , 0–Invalid |
| Latitude | 13.0965700 | Latitude in degrees |
| Longitude | 80.2913900 | Longitude in degrees |
| Altitude | 10.4 | Altitude in meters |
| Speed | 0.36 | Speed in Km/s |
| Direction | 128.43 | Direction in Degrees |
| Odometer | 2.500 | GPS, Cumulative Distance travelled in KM |
| GPS Move Status | 1 | 0-Stop,1-Moving,2-IDLE based on GPS |
| External Battery Volt | 12.6 | Vehicle Battery Volt |
| Internal Battery Volt Battery Volt \| State | 4.2\|1 | BattVolt \| State[ 0-disc,1-charging,2-full,3-discharging] |
| GSM Signal | 0-31 | GSM Signal in db |
| LAC | 63 | Location area code |
| Cell-ID | 3EAF | Cell Id |
| MCC | | Mobile country code |
| MNC | | Mobile network code |
| Alert Msg Code Code \| data | AA\|0 | Device Alert Message Events |
| Sensor Interface | 0 | Sensor Information 0-No interface,TS\|xx.xx |
| IGN | 1 | Ignition status 1-ON,0-OFF |
| Analog Input1 | 2.5 | Analog Input1 voltage |
| Digital Input1 | 1 | Digital input1 state: 1-ON,0-OFF |
| Output1 | 1 | Output1 state: 1-ON,0-OFF |
| Sequence Number | 000 | Sequence number |
| CheckSum | 4A# | Checksum of data packet from $ to # |

## *1.2 SENSOR INTERFACE:*

If sensor is not interfaced this field will be set to zero.
If Temperature sensor is connected then the field will start at ( "TS| temperature") code with separator to indicate the type of sensor followed by the temperature value in Celsius.
TS|xx.xx

Example:
TS|32.56

## *1.2 ALERT MESSAGE EVENTS:*

| Alert Format | Example | Alert Code | Data | Event Description |
|---|---|---|---|---|
| AA\|0 | AA\|0 | AA | 0 | No Event/Alarm |
| SO\|0 | SO\|0 | SO | 0 | Alert SOS Key Press |
| PU\|0 | PU\|0 | PU | 0 | Device Power UP |
| PF\|0 | PF\|0 | PF | 0 | Main battery to device Power Cut |
| PR\|0 | PR\|0 | PR | 0 | Main battery to device Power Restore |
| BO\|0 | BO\|0 | BO | 0 | Internal Battery ON |
| BL\|0 | BL\|0 | BL | 0 | Internal LOW Battery |
| BS\|0 | BS\|0 | BS | 0 | Internal LOW Battery Shut |
| IN\|0 | IN\|0 | IN | 0 | Ignition ON |
| IR\|0 | IR\|0 | IR | 0 | Ignition OFF |
| TU\|0 | TU\|0 | TU | 0 | Over Speed |
| II\|0 | II\|0 | II | 0 | Change in Digital Input State |
| OO\|0 | OO\|0 | OO | 0 | Change in Digital Output State |
| GE\|Zone | GE\|Beach Station | GE | x | Geo Zone Entry x-Zone Name max 48 char |
| GX\|Zone | GX\|Parrys | GX | x | Geo Zone Exit x-Zone Name max 48 char |
| GV\|x | GV\|1 | GV | 1 | GeoFence Violation x= 1 – IN, 0- OUT |
| IM\|0 | IM\|0 | IM | 0 | Immobilization ON |
| VI\|x | VI\|15 | VI | 15 | Vehicle in Idle, x- Idle stop time in min |
| VM\|0 | VM\|0 | VM | 0 | Vehicle in Motion |
| VS\|0 | VS\|0 | VS | 0 | Vehicle in Stop |
| TW\|0 | TW\|0 | TW | 0 | Tow Alarm |
| GF\|0 | GF\|0 | GF | 0 | GPS Communication Fault |
| FT\|0 | FT\|0 | FT | 0 | OTA Firmware Upgrade OK |
| FF\|0 | FF\|0 | FF | 0 | OTA Firmware Upgrade Failed |
| | | | | |
| | | | | |

## *1.2 DEVICE CONFIG READ/WRITE VIA TCP/HTTP COMM:*

This is the special feature of the device is provided to command, configure and read the settings of the device through tcp/http from backend server.

General command format to device read and write
**Send command from Server**:
$IPCFG,Command

**Reply from Device**:
$IPCFG,IMEI,Reply

$IPCFG is the header part of the command. Command specifies the type of command to be send. Please Refer Device command manual for detailed commands.

**TCP Server :**

1. If Communication Protocol is Selected as TCP then Send as

Example:
To query the Server IP Configuration details send <Get.ip> command.
**Command from Server**:
$IPCFG,<Get.ip>

**Reply from Device**:
$IPCFG, 863071013858334, <Cfg.ip: RMODE="1",PRI-IP="11.198.172.123",PRI-PORT="5001",SEC-IP="0.0.0.0",SEC-PORT="0",APN="airtelgprs.com",USER="",PSWD="" >

**HTTP Server:**

2. If Communication Protocol is selected as Http then data is enclosed in the body of the html part.

Example:
<body>
$IPCFG, <Get.ip>
</body>

**Reply from Device**:
The device will push the data by call back the http URL configured in the device.

http://11.198.172.123:8080/Track/eCallback?reply=$IPCFG, 863071013858334, <Cfg.ip: RMODE="1",PRI-IP="118.139.162.123",PRI-PORT="5001",SEC-IP="0.0.0.0",SEC-PORT="0",APN="airtelgprs.com",USER="",PSWD="" >

## *1.2 ADVANCE HTTP DEVICE API CALLBACK:*

This generic feature act as a bridge between the user sms query and the backend server to process the user commands directly. The reply send back to user via the device.

User can directly query Server information via from device using api callback feature.

The http Server callback url has to configured in the device
Example:
http://11.198.172.123:8080/Track/uCallback

User Can Send SMS with the query paramer as below
SMS *Password#<GET: qtag=qvalue >

Example:
*1111#<GET: status=locate >

On receiving above command by sms from the user . the device will start call the api url with the user query parameters.

Example:
http://11.198.172.123:8080/Track/uCallback? status=locate &format=json

The server response should be in json format. The data should be embed in the html body part of the reply.

{“api_reply”:”message part” }

The message part as send as sms to the user. Like wise the user can read or modify the settings and read the status from the server using the callback api.

## *1.2 SEND SMS COMMAND FROM SERVER:*

From server, Issue a Device command to Send SMS to the user by the Predefined format.

$IPCFG,<DEVCMD: SMS=phno,Message >

Phno : User Mobile Number
Message : Text Message upto 160 Character