

## COMMUNICATION PROTOCOL WIALON IPS v.2.0

Version	Date	Changes
2.0	10.2014	<p>In packet L (login packet) added the "protocol version."</p> <p>In packets L, SD, D, M, I, IT, T, US, UC added a checksum crc16.</p> <p>For packets AL, ASD, AD, AM, AI, AIT, AT added error code checksum.</p>

### Incoming TCP data protocol

All the data comes in text format and represent the following type packet:

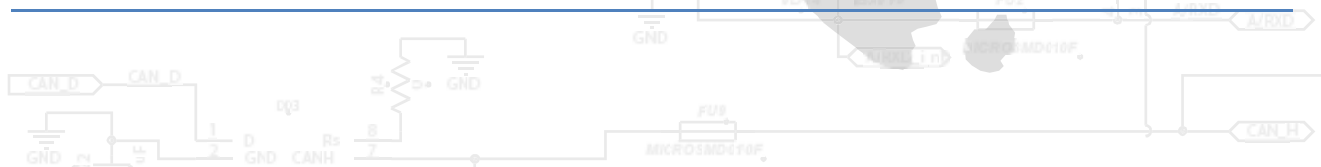
#TP#msg\r\n

#	start byte
TP	the type of packet, description of all possible types is shown in Table. 1
#	separator
msg	message
\r\n	end of packet

### The types of packet

(table 1)

Type	Description	Sender
L	packet login	equipment
AL	response to the packet login	server
D	data packet	equipment
AD	response to the data packet	server
P	ping packet	equipment
AP	response to the ping packet	server
SD	short data packet	equipment
ASD	response to the short data packet	server
B	packet with black box	equipment
AB	response to the packet with black box	server
M	message to the driver	equipment / server
AM	response to the message from the driver	server
US	packet with new firmware	equipment
UC	packet with configuration file	server



## Login packet

#L#protocol\_version;imei;password;crc16\r\n

<b>;</b>	separator
<b>protocol_version</b>	Version of protocol. Field must contain a value 2.0
<b>imei</b>	unique tracker ID, IMEI or serial number
<b>password</b>	password to device access, if empty - NA
<b>crc16</b>	checksum (Appendix 1)

In response to login packet server sends a command to AL:

"1" - if successful authorization to the server;

"0" - if no connection with server;

"01" - error checking password;

"10" - if checksum error.

Examples:

#AL#1\r\n      #AL#0\r\n

## Short data packet

#SD#date;time;lat1;lat2;lon1;lon2;speed;course;height;sats;crc16\r\n

<b>date</b>	date in format DDMMYY, in UTC, if empty - NA
<b>time</b>	time in format HHMMSS, in UTC, if empty - NA
<b>lat1;lat2</b>	latitude (5544.6025;N), if empty - NA;NA
<b>lon1;lon2</b>	longitude (03739.6834;E), if empty - NA;NA
<b>speed</b>	speed, integer, km/h, if empty - NA
<b>course</b>	course, integer, degrees, if empty - NA
<b>height</b>	height, integer, in meters, if empty - NA
<b>sats</b>	number of satellites, integer, if empty - NA
<b>crc16</b>	checksum (Appendix 1)

If the date and time contain NA, the current server time used.

In response to the short data packet server sends ASD:

"-1" - error package structure

"0" - incorrect time

"1" - packet successfully fixed

"10" - error to get coordinates

"11" - error to get the speed, course or altitude

"12" - error to get the number of satellites

"13" - checksum error.

Examples:

#ASD#1\r\n      #ASD#0\r\n      #ASD#10\r\n      #ASD#11\r\n      #ASD#12\r\n

## Data packet

#D#date;time;lat1;lat2;lon1;lon2;speed;course;height;sats;hdop;inputs;outputs;adc;ibutton;params;crc16\r\n

<b>date</b>	date in format DDMMYY, in UTC, if empty - NA
<b>time</b>	time in format HHMMSS, in UTC, if empty - NA
<b>lat1;lat2</b>	latitude (5544.6025;N), if empty - NA
<b>lon1;lon2</b>	longitude (03739.6834;E), if empty - NA;NA
<b>speed</b>	speed, integer, km/h, if empty - NA
<b>course</b>	course, integer, degrees, if empty - NA
<b>height</b>	height, integer, in meters, if empty - NA
<b>sats</b>	number of satellites, integer, if empty - NA
<b>hdop</b>	decline of precision, fractional number, if empty - NA
<b>inputs</b>	digital inputs, each bit number corresponds to one input, starting with the youngest, integer, if empty - NA
<b>outputs</b>	digital outputs, each bit number corresponds to one outputs, starting with the youngest, integer, if empty - NA
<b>adc</b>	analog inputs, fractional numbers, separated by a comma. The numbering starts with the output unit; If no analog inputs - transferred to an empty string.  Exempl: 14.77,0.02,3.6
<b>ibutton</b>	code key driver string of arbitrary length. If no key is passed NA
<b>params</b>	set of additional parameters separated by a comma.  Each parameter represents a construction NAME:TYPE:VALUE NAME - arbitrary string of length at most 15 bytes; TYPE - parameter type, 1 -int/long long, 2 -double, 3 -string VALUE - value depending on the type  The first type parameter with name "SOS" used to send alarm button, value 1 means the button is pushed.  The third parameter type (string) with name "text" used to send a text message. This parameter can be used to send a text message from driver, which may contain the coordinates and other parameters.  Examples: count1:1:564,fuel:2:45.8,hw:3:V4.5  SOS:1:1
<b>crc16</b>	checksum (Appendix 1)

If the date and time contain NA, the current server time used.

In response to the packet data server sends AD:

- "-1" - error package structure
- "0" - incorrect time
- "1" - packet successfully fixed
- "10" - error to get coordinates
- "11" - error to get the speed, course or altitude



## Packet with new firmware

Used to send new firmware to the tracker.

#US#sz;crc16\r\nBIN

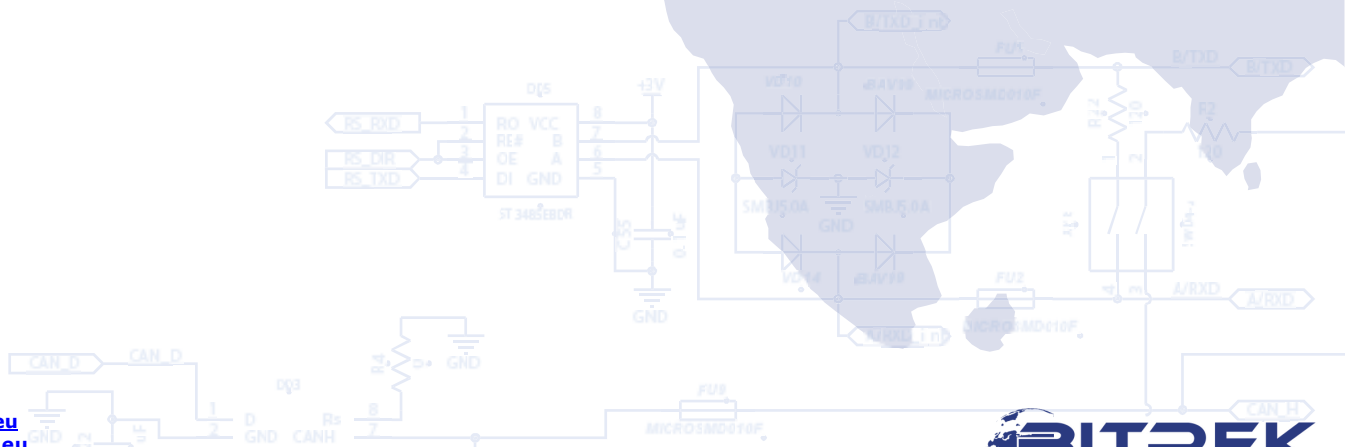
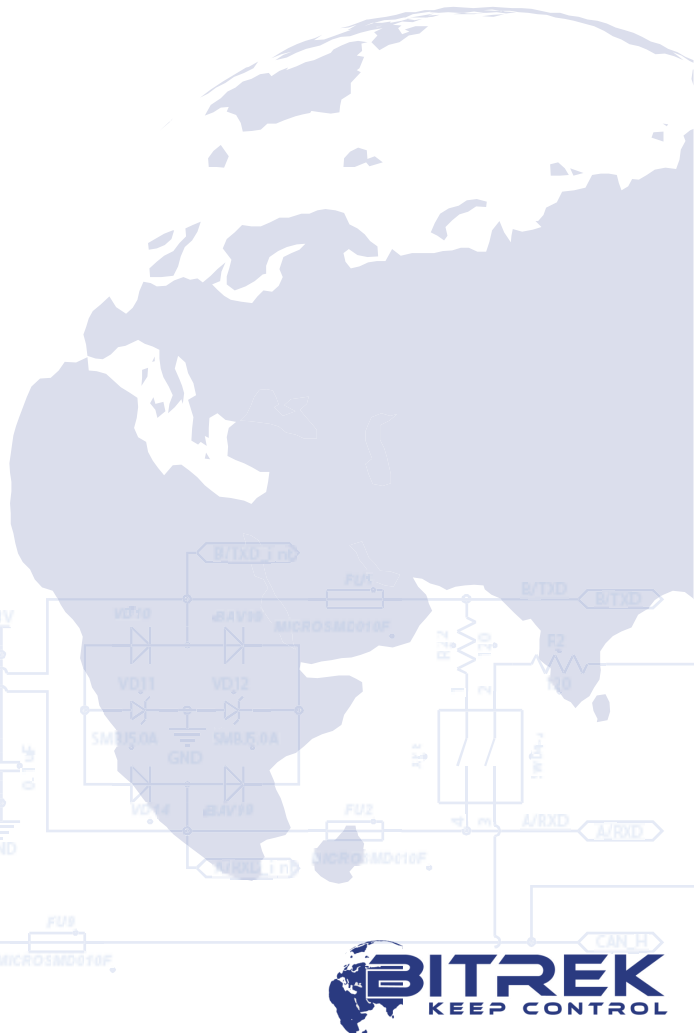
<b>sz</b>	size of binary data packet (for example, 51200 bytes)
<b>crc16</b>	checksum (Appendix 1)
<b>BIN</b>	firmware in binary form

## Packet with configuration file

Used to send configuration file to the tracker

#UC#sz;crc16\r\nBIN

<b>sz</b>	size of the configuration file, byte
<b>crc16</b>	checksum (Appendix 1)
<b>BIN</b>	contents of the configuration file



Field `crc16` must contain the checksum value in hexadecimal format big-endian, without leading zeros on the left, for example:

AA13BB, that represents a decimal format number 11146171.

In packets L, SD, D, B, M for calculation `crc16` taken part of the packet between # TP # and field `crc16`.

Packet example:

```
#SD#date;time;lat1;lat2;lon1;lon2;speed;course;height;sats;crc16\r\n
```

In this case `crc16` calculated for the next portion of the packet:

```
date;time;lat1;lat2;lon1;lon2;speed;course;height;sats;
```

Packet example:

```
#B#date;time;lat1;lat2;lon1;lon2;speed;course;height;sats|date;time;lat1;lat2;lon1;lon2;speed;course;height;sats|crc16\r\n
```

In this case `crc16` calculated for the next portion of the packet:

```
date;time;lat1;lat2;lon1;lon2;speed;course;height;sats|date;time;lat1;lat2;lon1;lon2;speed;course;height;sats|
```

In packets I, US, UC, T for calculation `crc16` taken field BIN.

Packet example:

```
#I#51200;0;1;070512;124010;sample.jpg;crc16\r\nBIN
```

In this case `crc16` calculated for the field BIN.

Code example in C language for `crc16` calculation:

```
static const unsigned short crc16_table[256] =
{
0x0000,0xC0C1,0xC181,0x0140,0xC301,0x03C0,0x0280,0xC241,0xC601,0x06C0,0x0780,0xC741,
0x0500,0xC5C1,0xC481,0x0440,0xCC01,0x0CC0,0x0D80,0xCD41,0x0F00,0xFCF1,0xCE81,0x0E40,
0x0A00,0xCAC1,0xCB81,0x0B40,0xC901,0x09C0,0x0880,0xC841,0xD801,0x18C0,0x1980,0xD941,
0x1B00,0xDBC1,0xDA81,0x1A40,0x1E00,0xDEC1,0xDF81,0x1F40,0xDD01,0x1DC0,0x1C80,0xDC41,
0x1400,0xD4C1,0xD581,0x1540,0xD701,0x17C0,0x1680,0xD641,0xD201,0x12C0,0x1380,0xD341,
0x1100,0xD1C1,0xD081,0x1040,0xF001,0x30C0,0x3180,0xF141,0x3300,0xF3C1,0xF281,0x3240,
0x3600,0xF6C1,0xF781,0x3740,0xF501,0x35C0,0x3480,0xF441,0x3C00,0xFCF1,0xFD81,0x3D40,
0xFF01,0x3FC0,0x3E80,0xFE41,0xFA01,0x3AC0,0x3B80,0xFB41,0x3900,0xF9C1,0xF881,0x3840,
0x2800,0xE8C1,0xE981,0x2940,0xEB01,0x2BC0,0x2A80,0xEA41,0xEE01,0x2EC0,0x2F80,0xEF41,
0x2D00,0xEDC1,0xEC81,0x2C40,0xE401,0x24C0,0x2580,0xE541,0x2700,0xE7C1,0xE681,0x2640,
0x2200,0xE2C1,0xE381,0x2340,0xE101,0x21C0,0x2080,0xE041,0xA001,0x60C0,0x6180,0xA141,
0x6300,0xA3C1,0xA281,0x6240,0x6600,0xA6C1,0xA781,0x6740,0xA501,0x65C0,0x6480,0xA441,
0x6C00,0xACC1,0xAD81,0x6D40,0xAF01,0x6FC0,0x6E80,0xAE41,0xAA01,0x6AC0,0x6B80,0xAB41,
0x6900,0xA9C1,0xA881,0x6840,0x7800,0xB8C1,0xB981,0x7940,0xBB01,0x7BC0,0x7A80,0xBA41,
0xBE01,0x7EC1,0x7F80,0xBF41,0x7D00,0xBDC1,0xBC81,0x7C40,0xB401,0x74C0,0x7580,0xB541,
0x7700,0xB7C1,0xB681,0x7640,0x7200,0xB2C1,0xB381,0x7340,0xB101,0x71C0,0x7080,0xB041,
0x5000,0x90C1,0x9181,0x5140,0x9301,0x53C0,0x5280,0x9241,0x9601,0x56C0,0x5780,0x9741,
0x5500,0x95C1,0x9481,0x5440,0x9C01,0x5CC0,0x5D80,0x9D41,0x5F00,0x9FC1,0x9E81,0x5E40,
0x5A00,0x9AC1,0x9B81,0x5B40,0x9901,0x59C0,0x5880,0x9841,0x8801,0x48C0,0x4980,0x8941,
0x4B00,0x8BC1,0x8A81,0x4A40,0x4E00,0x8EC1,0x8F81,0x4F40,0x8D01,0x4DC0,0x4C80,0x8C41,
0x4400,0x84C1,0x8581,0x4540,0x8701,0x47C0,0x4680,0x8641,0x8201,0x42C0,0x4380,0x8341,
0x4100,0x81C1,0x8081,0x4040
};
unsigned short crc16 (const void *data, unsigned data_size)
{
if (!data || !data_size)
return 0;
unsigned short crc = 0;
unsigned char* buf = (unsigned char*)data;
while (data_size--)
crc = (crc >> 8) ^ crc16_table[((unsigned char)crc ^ *buf++]);
return crc; }

```