

Aaronia GPS Logger Programming Guide



Introduction

The Aaronia GPS Logger is a battery-powered mobile device to measure and record location and orientation related information from a multitude of sensors:

- Position and time from a GPS receiver
- Orientation from a electronic compass (magnetometer) and accelerometer
- Rotation information from a gyroscope
- Pressure from a barometer which may be used with other sources to obtain elevation information

Above data can be recorded directly on a inserted Micro-SD card, or transmitted to a connected system over an integrated USB interface.

This document provides the necessary details about the communication protocol, data formats and processing steps to develop custom applications for the Aaronia GPS Logger.

USB Serial Connection

Communication with the GPS Logger is performed over a USB serial connection. The device uses a FTDI USB-serial interface chip, so to establish a connection you must use the corresponding FTDI drivers, which are included and automatically installed using the Aaronia MCS software installer or the standalone Aaronia driver package (note: on Windows, stock FTDI drivers will not work due to different PID configuration). As the protocol is text-based you can use either the low-level D2XX driver library or the high-level VCP terminal interface with the following settings:

- Baud Rate: 625.000
- Word Length: 8
- Stop Bits: 2
- Parity Bits: 0
- Flow Control: 0
- Xon: 0
- Xoff: 0

Other settings may work but are not supported.

For details on how to use the respective driver interfaces please refer to the documentation available on the FTDI website: <http://www.ftdichip.com/FTSupport.htm>

SD Card Specifics

The GPS Logger can record data on a inserted Micro-SD card. This can be triggered by software (see MODE) or by enabling the „Logger“ switch on the device itself.

In order to be used by the GPS Logger a SD card must be formatted with the FAT16 filesystem. Other filesystems are not supported, including FAT32, exFAT and NTFS. Depending on the system used to format it the maximum usable size of the card is limited to 512 MB to 4 GB and the maximum number of files on the card to about 500. Also filenames are restricted to the 8.3 naming convention.

Communication Protocol

The GPS Logger uses the text-based NMEA 0183 syntax for communication, including the standard GPRMC and GPGLGA sentences. Commands and sensor-specific replies use the PAAG vendor suffix. In general a command has the form

`$PAAG, <command>, <parameters><CR><LF>`

and replies use

`$<replycode>, <parameters>* <checksum><CR><LF>`

where CR and LF are ASCII codes 13 and 10, and checksum is a 2 character hexadecimal representation of the XOR combination of all characters between the \$ and the * (as defined by the NMEA protocol).

Commands may trigger zero, one or multiple replies.

The Aaronia GPS Logger implements the following commands (see Data Format for replies):

ID

- Check Hardware, Firmware and Protocol versions
- No parameters
- Reply: PAAG, ID
- Example: `$PAAG, ID<CR><LF>`

MODE

- Set data streaming mode
- One parameter with values:
 - START: enable continuous streaming
 - STOP: disable streaming
 - READONE: stream one record
- Reply: GPRMC, GPGLGA and PAAG, DATA (in unspecified order)
- Example: `$PAAG, MODE, START<CR><LF>`

FILE

- Control file operations on inserted SD card
- Two parameters, <action> and <param> as follows:
 - action is START: start logging into file with number given in param
 - action is STOP: stop logging, param is empty string
 - action is LIST: list recorded files, param is first file number to be listed
 - action is DUMP: stream logged content of file with number given in param
 - action is STAT: report file size and date of file with number given in param
 - action is DEL: delete file with number given in param from SD card
- Reply: no reply for actions START, STOP and DEL;
 - for action LIST: PAAG, FILELIST
 - for action DUMP: GPRMC, GPGGA and PAAG, DATA
 - for action STAT: PAAG, FILE, STAT
- Example: \$PAAG, FILE, LIST, 11<CR><LF>

Data Format

All data coming from the GPS Logger is reported in the form of NMEA 0183 sentences, see Communication Protocol for details. By default data streaming is disabled, you need to issue a command to get any replies (usually PAAG, MODE, START). Generated logfiles on the SD card use the same format and contain only replies, no commands, so the same parser can be used for the serial interface and file contents.

Even though the format is text-based it operates on a low level, so values of data replies correspond to raw sensor values that usually aren't fit to be used in applications directly. See Processing Raw Data for further information how to obtain usable values.

The GPS Logger implements the following replies:

GPRMC

- Reports date, time, GPS coordinates and estimated speed and course
- Has 12 parameters, which may be empty/invalid on missing signal:
 1. Time: Format is HHMMSS.ZZZ (ZZZ being milliseconds, but is always zero)
 2. Status: A for ok, V for warnings
 3. Latitude: Float value, see XXXX for details
 4. Latitude orientation: N for north, S for south
 5. Longitude: Float value, see XXXX for details
 6. Longitude orientation: W for west, E for east
 7. Speed over ground: Float value as knots (estimated value based on GPS data)
 8. Course: Float value as heading over ground relative to true north (estimated value based on GPS data)
 9. Date: Format is DDMMYY (no century information)
 10. Magnetic deviation: empty
 11. Magnetic deviation orientation: empty
 12. Signal integrity: A for valid signal, N for invalid or no signal
- Example:

```
$GPRMC,111529.000,A,5008.2031,N,00619.1924,E,0.33,346.82,120213,,A*64<CR><LF>
```

GPGGA

- Reports time, GPS coordinates, elevation and details on satellite signal
- Has 14 parameters, which may be empty/invalid on missing signal:
 1. Time (see GPRMC)
 2. Latitude (see GPRMC)
 3. Latitude orientation (see GPRMC)
 4. Longitude (see GPRMC)
 5. Longitude orientation (see GPRMC)
 6. GPS quality: 0 for invalid, 1 for having a GPS fix
 7. Number of satellites
 8. Horizontal deviation
 9. Elevation: Height of Antenna over sea level
 10. Elevation unit: M for meters
 11. Geoidal separation
 12. Geoidal separation unit: M for meters
 13. Age of DGPS record: empty
 14. DGPS reference: empty
- Example:
`$GPGGA,111529.000,5008.2031,N,00619.1924,E,1,6,1.45,414.4,M,47.7,M,,*59<CR><LF>`

PAAG,DATA

- Reports data of the non-GPS sensors
- Has 6 parameters:
 1. Sensor: G for gyroscope, C for compass, B for barometer, T for accelerometer
 2. Time: Format is HHMMSS.NNN, where NNN is an increasing counter that is reset when the time is updated by a new GPRMC or GPGGA record
 3. Value of sensor X-axis (see Processing Raw Data)
 4. Value of sensor Y-axis, empty for barometer (see Processing Raw Data)
 5. Value of sensor Z-axis, empty for barometer (see Processing Raw Data)
 6. Status: A for ok, N for invalid
- Example:
`$PAAG, DATA, T, 145802.0, 2113, -63, 8257, A*0B<CR><LF>`

PAAG,ID

- Reports hardware, firmware and protocol version
- Has 3 parameters:
 1. Hardware version
 2. Firmware version
 3. Protocol version
- Example: `$PAAG, ID, 1, 1, 1*2B<CR><LF>`

PAAG,FILELIST

- Reports a list of existing logfile numbers
- Has 9 parameters:
 1. Number to pass to PAAG, FILE, LIST to get the next group of files, or empty if no further files exist
 2. Number of an existing file, or empty
 3. Number of an existing file, or empty
 4. Number of an existing file, or empty
 5. Number of an existing file, or empty
 6. Number of an existing file, or empty
 7. Number of an existing file, or empty
 8. Number of an existing file, or empty
 9. Number of an existing file, or empty
- Example: \$PAAG, FILELIST, 25, 11, 13, 16, 17, 18, 20, 23*31<CR><LF>

PAAG,FILE,STAT

- Reports file size and modification date
- Has 4 parameters:
 1. File number
 2. Size in bytes
 3. Date: Format is DD.MM.YY (no century information)
 4. Time: Format is HH:MM (no seconds or milliseconds)
- Example: \$PAAG, FILE, STAT, 3, 473978, 12.06.13, 11:56*08

Processing Raw Data

Due to limited processing power in the GPS Logger device itself most of the data values are taken directly from the sensors without any further processing. These values are often not suitable for direct usage as they are not in the usually expected units and/or scale.

GPS Coordinates

The GPS sensor transmits coordinates as WGS84 „degree – minutes - hemisphere“ encoded in a floating point number and suffix in the format DDMM.MMMM,H. Most electronic systems expect coordinates in decimal degrees. The conversion formula is as follows:

$$\text{DecimalDegree} = (\text{DD} + \text{MM.MMMM} / 60)$$

(and values in western / southern hemisphere have to be negated)

Example:

$$5008.2032,N = 50 + 8.2032 / 60 = 50.13672$$

Accelerometer values

Converting accelerometer readout values to tilt values in degrees is a complex multi-step process, so only a C implementation is provided here:

```
double x    = tilt.x / 8192;
double y    = tilt.y / 8192;
double z    = tilt.z / 8192;
double d_pi = 180.0 / M_PI;
result.x = -atan2(y, sqrt((x * x) + (z * z))) * d_pi;
result.y =  atan2(-x, (z < 0 ? -1 : 1) * sqrt((y * y) + (z * z))) * d_pi;
result.z = 0;
```

Barometer Values

The barometer reports the current pressure in hectopascal (hPa). This can be used directly.

Gyroscope Values

To obtain gyroscope values as degree/s you have to divide the readout values of each axis by 14.375.

Example:

$$117 = 117 / 14.375 = 8.1391^\circ/\text{s}$$

Compass Values

Converting compass readout values to a heading value requires multiple steps. First you have to normalize the values to Gauss by dividing each axis value by 1090. This gives you the magnetic field strength of each direction.

Example:

$$\text{Compass} = (209, -1, -404)$$

$$\text{CompGauss} = (209/1090, -1/1090, -404/1090)$$

$$\text{CompGauss} = (0.1917 \text{ Gauss}, -0.001 \text{ Gauss}, -0.3706 \text{ Gauss})$$

A simple heading calculation can be done by taking the arcus tangens of the x- and y-axes, and add a half-circle if the result is negative.

$$\text{HeadingRadian} = \arctan(-1/209) = -0.7854$$

(Implementation note: when available use $\text{atan2}(y, x)$, else you may have to compensate the result for negative values of x and check for $x = 0$ before the division)

$$\text{HeadingRadian} = -0.7854 + 2 * \pi = 5.4978$$

$$\text{HeadingDegree} = 5.4978 * 180 / \pi = 315.001^\circ$$