Before we begin

By using the following documentation to read and write commands to your RaceBox Mini, RaceBox Mini S, or RaceBox Micro you agree that RaceBox Llc is not responsible for any damage caused to you or your RaceBox produced by incorrect commands or configuration. Warranty claims will be void.

Preface

RaceBox Mini, Mini S, and Micro include a GNSS receiver, Gyroscope, and Accelerometer. The data from the sensors is combined and provided using UART-over-BLE. The following document aims to provide basic information on receiving information from the device. It applies to RaceBox Mini, RaceBix Mini S, and RaceBox Micro firmware versions 2.x and 3.x. There are no breaking changes from 1.x to 2.x or from 2.x to 3.x. Apps, built around older versions will work out of the box but may not have access to the new features.

Identifying a RaceBox Device Model

When scanning for BLE devices a client can detect a RaceBox BLE device using a name and advertised services filter. The name should always start with “RaceBox Mini “, “RaceBox Mini S “ or “RaceBox Micro “ and a unique 10-digit serial number.

After the connection is established the client can further verify the device’s version and manufacturer using the Device Information Service. All characteristics in the service contain read-only string data

Device Info Service UUID	0000180a-0000-1000-8000-00805f9b34fb
Model Characteristic UUID	00002a24-0000-1000-8000-00805f9b34fb
Serial Number Characteristic UUID	00002a25-0000-1000-8000-00805f9b34fb
Firmware Revision Characteristic UUID	00002a26-0000-1000-8000-00805f9b34fb
Hardware Revision Characteristic UUID	00002a27-0000-1000-8000-00805f9b34fb
Manufacturer Characteristic UUID	00002a29-0000-1000-8000-00805f9b34fb

The Model Characteristic will contain either the string “RaceBox Mini”, “RaceBox Mini S” or “RaceBox Micro” depending on the device. The firmware revision characteristic will contain a string in the format “major.minor”. For example “2.6”. These two characteristics can be used to identify the device model and what commands are available.

UART-over-BLE

RaceBox Mini, RaceBox Mini S, and RaceBox Micro provide a UART service with an RX and TX characteristic. The relevant UUIDs are:

UART Service UUID	6E400001-B5A3-F393-E0A9-E50E24DCCA9E
RX characteristic UUID	6E400002-B5A3-F393-E0A9-E50E24DCCA9E
TX characteristic UUID	6E400003-B5A3-F393-E0A9-E50E24DCCA9E

Clients can write commands to the RX characteristic and subscribe for notifications of the TX characteristic to obtain command responses and unsolicited periodic messages. It is enough to subscribe and parse the default data messages for basic implementations.

Devices running firmware 3.3 and later provide additional UART service for polling live location data in standard NMEA protocol:

NMEA UART Service UUID	00001101-0000-1000-8000-00805F9B34FB
NMEA RX characteristic UUID	00001102-0000-1000-8000-00805F9B34FB
NMEA TX characteristic UUID	00001103-0000-1000-8000-00805F9B34FB

While the NMEA UART has both RX and TX characteristics it is read-only. Data sent to the RX characteristic will be ignored. It exists only for compatibility reasons.
It is NOT recommended to use both RaceBox Protocol and NMEA. Sending both protocols data over BLE will result in high packet loss.

MTU and connection intervals

Most modern smartphones and computers support BLE Data Length Extension. RaceBox devices send a lot of data and the client must be able to handle it. Setting the connection interval as low as possible and the MTU as high as possible is recommended.

To accommodate clients that can’t provide high-enough MTUs the device will split RaceBox Protocol packets into multiple notifications. Clients must be able to buffer the data and reassemble them or they will lose the packets’ contents. Do NOT assume that each BLE notification contains only one message or that it contains a complete message. That applies to the process of writing data to the device as well. However, for the NMEA protocol, the device will not split messages and send each as exactly one BLE characteristic update to maintain compatibility with some existing applications.

Packets Format

Packets are sent and received in U-Blox UBX binary format, consisting of a header, optional payload, and a checksum. The header is used to identify the packet start, contents, and payload length. The checksum is used to verify the packet transmission.

Data in the packet is encoded in Little-Endian and uses only integer fields - all double values are multiplied with some value and rounded to the nearest integer. Up to 32-bit values are used.

The structure is as follows:

2 bytes Packet start	2 bytes Message class and I	2 bytes Payload length	Payload Variable Length	2 bytes Checksum

- The first 2 bytes are the frame start - always 0xB5 and 0x62.
- The two bytes for message class and ID identify the packet contents - the first byte is the class, and the second is the ID
- The two bytes for payload length should be read as an unsigned 2-byte integer and indicate the payload size. It can be from 0 to 504 bytes. The UBX protocol allows for larger messages, but the device's internal buffers limit the full packet size to 512 bytes.
- The 2-byte checksum is calculated over the packet's contents - the message class and ID bytes, the payload length bytes, and the payload itself. The formula is:

```
// Assuming Packet is a byte array containing the entire
// packet with a header and 2 spare bytes at the end for
// the checksum:
byte CK_A = 0, CK_B = 0
for (int i = 2; i < len(Packet)-2; i++) {
      CK_A = CK_A + Packet[i]
      CK_B = CK_B + CK_A
}
Packet[len(Packet)-2] = CK_A
Packet[len(Packet)-1] = CK_B
```

Clients must validate the checksum when receiving a packet to ensure it has not been damaged in transport.

As noted before one must not rely on a single BLE notification containing exactly one full packet. Developers must buffer the data to assemble packets that may have been split into multiple BLE characteristic notifications. Verify that the buffered data starts with the appropriate header, contains enough data, and that the checksum matches before parsing the packet.

RaceBox Data Message

Message class: 0xFF
Message ID: 0x01
Payload size: 80 bytes

The device sends up to 25 times per all important data - GNSS Location, Accelerometer, Gyroscope, and others in one single message. It is enabled immediately on connection and you can start receiving it as soon as you subscribe for the BLE characteristic notifications. It is derived from the U-Blox NAV-PVT message with some fields removed and some added. The payload is 80 bytes:

Offset	Size	Type	Description
0	4 bytes	UInt32	iTOW
4	2 bytes	UInt16	Year
6	1 byte	Byte	Month
7	1 byte	Byte	Day
8	1 byte	Byte	Hour
9	1 byte	Byte	Minute
10	1 byte	Byte	Second
11	1 byte	Bitmask	Validity Flags
12	4 bytes	UInt32	Time Accuracy
16	4 bytes	Int32	Nanoseconds
20	1 byte	Enum	Fix Status
21	1 byte	Bitmask	Fix Status Flags
22	1 byte	Bitmask	Date/Time Flags
23	1 byte	Byte	Number of SVs
24	4 bytes	Int32	Longitude
28	4 bytes	Int32	Latitude
32	4 bytes	Int32	WGS Altitude
36	4 bytes	Int32	MSL Altitude
40	4 bytes	UInt32	Horizontal Accuracy

44	4 bytes	UInt32	Vertical Accuracy
48	4 bytes	Int32	Speed
52	4 bytes	Int32	Heading
56	4 bytes	UInt32	Speed Accuracy
60	4 bytes	UInt32	Heading Accuracy
64	2 bytes	UInt16	PDOP
66	1 byte	Bitmask	Lat/Lon Flags
67	1 byte	Bitmask	Battery Status or Input Voltage
68	2 bytes	Int16	GForceX
70	2 bytes	Int16	GForceY
72	2 bytes	Int16	GForceZ
74	2 bytes	Int16	Rotation rate X
76	2 bytes	Int16	Rotation rate Y
78	2 bytes	Int16	Rotation rate Z

Fields description

A lot of the fields are directly copied from the U-Blox NAV-PVT message. Details about their meaning can be found in the U-Blox Interface Description.

- iTOW: number of milliseconds from the GPS week start
- Year, month, day, hour, minute, second, and nanosecond form UTC timestamp of the message. Note that the Nanoseconds are signed and can be negative. Month indexing starts from 1 for January
- Fix Status: indicates the location solution status. Possible values:
    - 0: no fix
    - 2: 2D fix
    - 3: 3D fix

  To determine whether the receiver has a good location solution it is recommended to check if bit 0 in the Fix Status Flags is set and the value of FixStatus is 3.
- Validity Flags

BIt 0	1 = valid date
Bit 1	1 = valid time
Bit 2	1 = fully resolved

Bit 3	1 = valid magnetic declination

- Time Accuracy: in nanoseconds
- Fix Status Flags

Bit 0	1 = valid fix
Bit 1	1 = differential corrections applied
Bits 4..2	Power State
Bit 5	1 = valid heading
Bits 7..6	Carrier phase range solution

- Date/Time Flags

Bit 5	1 = Available confirmation of Date/Time Validity
Bit 6	1 = Confirmed UTC Date Validity
Bit 7	1 = Confirmed UTC Time Validity

- Number of SVs: the number of Space Vehicles used to compute the solution
- Longitude and Latitude: coordinates of the receiver with a factor of 10^7
- WGS and MSL altitude: Altitude in millimeters. The WGS altitude is in the Ellipsoid coordinate system, while MSL is an approximation of the altitude above the Mean Sea Level. Note that the GNSS receiver has a limited map of the Earth's gravitational field and for best results, it is recommended to implement the client-side conversion of WGS to MSL altitude.
- Horizontal and Vertical Accuracy: indication of the receiver's location error in millimeters.
- Speed: the speed of the vehicle in millimeters per second. For firmware before 3.3 this is always ground speed. For firmware 3.3 and later this can be switched between ground and 3D speed (see GNSS Platform Configuration)
- Heading: the direction of motion in degrees with a factor of 10^5, where zero is North
- Speed accuracy: estimation of the error of the Speed field in millimeters per second
- Heading Accuracy: estimation of the error of the Heading fid in degrees with a factor of 10^5
- PDOP - Position Dilution of Precision - indicates the error propagation of the satellite configuration. Usually directly related to the number of satellites. Value is with a factor of 100.
- Lat/Lon Flags

Bit 0	1 = Invalid Latitude, Longitude, WGS Altitude, and MSL Altitude
Bits 4..1	Differential Correction Age

- Battery Status or Input Voltage: for RaceBox Mini and Mini S this field contains charging status in the most significant bit (1 if charging) and estimation of the battery level in percentage in the remaining 7 bits. For RaceBox Micro this field contains the input voltage, multiplied by 10 (e.g. value of 0x79 means 12.1V)
- GForce X, Y, and Z: acceleration on the 3 axes in milli-g. Divide by a factor of 1000 to convert to g values. The orientation of the axes is X - front/back, Y - right/left, Z - up/down

- Rotation Rate X, Y, and Z: rotation speed on the 3 axes in centi-degrees per second. Divide by a factor of 100 to convert to degrees per second. The orientation of the axes is X - roll, Y - pitch, and Z - yaw. See below for the full diagram.

Basic client implementations do not need to handle the data in all the fields. Constructing proper UTC Date/Time of the packet (or using iTOW as a time interval between packets), parsing the receiver's Fix Status Flags, location (Latitude, Longitude, and Altitude), and attitude (Speed, Heading, G-Forces, and Rotation Rates) is usually enough. The remaining accuracy fields, flags, etc are intended mostly for monitoring the device's status.

Example packet from RaceBox Mini / Mini S

Sent from the device periodically (up to 25 times a second). The example is in hexadecimal form and contains headers and a checksum.

```
B5 62 FF 01 50 00 A0 E7 0C 07 E6 07 01 0A 08 33
08 37 19 00 00 00 2A AD 4D 0E 03 01 EA 0B C6 93
E1 0D 3B 37 6F 19 61 8C 09 00 0F 01 09 00 9C 03
00 00 2C 07 00 00 23 00 00 00 00 00 00 00 D0 00
00 00 88 A9 DD 00 2C 01 00 59 FD FF 71 00 CE 03
2F FF 56 00 FC FF 06 DB
```

Decoding:

Offset	Contents	Field	Decoded
0	B5 62	Header	
2	FF 01	Message Class and ID	RaceBox Mini Data Message
4	50 00	Payload Length	80 bytes
6	A0 E7 0C 07	iTOW	118286240
10	E6 07	Year	2022
12	01	Month	January
13	0A	Day	10th
14	08	Hour	8
15	33	Minute	51
16	08	Second	8
17	37	Validity Flags	Date/Time are valid and fully resolved

18	19 00 00 00	Time Accuracy	25 ns
22	2A AD 4D 0E	Nanoseconds	239971626ns = 0.239 seconds
26	03	Fix Status	3D Fix
27	01	Fix Status Flags	GNSS Fix OK
28	EA	Date/Time Flags	Date/Time Confirmed
29	0B	Number of SVs	11 SIVs
30	C6 93 E1 0D	Longitude	23.2887238
34	3B 37 6F 19	Latitude	42.6719035
38	61 8C 09 00	WGS Altitude	625.761 meters
42	0F 01 09 00	MSL Altitude	590.095 meters
46	9C 03 00 00	Horizontal Accuracy	0.924 meters
50	2C 07 00 00	Vertical Accuracy	1.836 meters
54	23 00 00 00	Speed	35mm/s = 0.126 kph
58	00 00 00 00	Heading	0 degrees
62	D0 00 00 00	Speed Accuracy	208 mm/s = 0.704 kph
66	88 A9 DD 00	Heading Accuracy	145.26856 degrees
70	2C 01	PDOP	3
72	00	Lat/Lon Flags	Coordinates are valid
73	59	Battery Status	89%, not charging
74	FD FF	GForceX	-0.003 g
76	71 00	GForceY	0.113 g
78	CE 03	GForceZ	0.974 g
80	2F FF	Rotation rate X	-2.09 deg/sec
82	56 00	Rotation rate Y	0.86 deg/sec
84	FC FF	Rotation rate Z	-0.04 deg/sec
86	06 DB	Checksum	Valid

When the device reports the battery is fully discharged (0%) a client should disconnect as soon as possible. The device will try to continue operating as long as it can until the battery is drained but it is not recommended to run it that far. Only if the battery reaches 0% while the client is recording important data should it maintain operation and disconnect otherwise.

GNSS Receiver Configuration

Message class: 0xFF
Message ID: 0x27
Payload size: 0-3 bytes

This configuration allows clients to change how the GNSS receiver processes data internally. It is supported by firmware 3.3 and later. Please note implementors should check with the protocol version to determine if this feature is available.

To read the current configuration send the message with an empty (zero bytes long) payload. The device will reply with a message of the same class and ID with a 3-byte payload with the following structure:

Offset	Size	Type	Description
0	1 byte	Byte	Dynamic Platform Model
1	1 byte	Boolean	Enable 3D-Speed reporting
2	1 byte	Byte	Minimum horizontal accuracy for a location fix

To change the configuration send the same command with a 3-byte payload with the same structure. The device will reply to the command with an ACK (0xFF 0x02) or NACK (0xFF 0x03) packet to confirm or reject the command. A NACK (0xFF 0x03) will be sent in the following conditions:

- The device does not support the command
- The specified Dynamic Platform Model is not valid (value range is 0 to 8)

Fields Description

- Dynamic Platform Model: configures an internal filter that improves the reported location and speed accuracy. The values correspond to the U-Blox CFG-NAVSPG-DYNMODEL definition. Recommended values are:
    - 4: for automotive use and generally any ground-based use case
    - 5: for sea use
    - 6: for airborne use with low dynamics (e.g. planes)
    - 8: for airborne use with high dynamics (e.g. racing drones) and vehicles that go above 300 kph

- Enable 3D-Speed reporting: if set to non-zero value the Speed field in the RaceBox Data Message will be 3d-speed instead of ground speed. For any ground-based usage, this should be set to 0 (ground speed)
- Minimum horizontal accuracy for a location fix: if the location accuracy is worse than the specified value (in meters) the receiver will report fix loss by setting bit zero of the Fix Status Flags to zero.

RaceBox products are configured with an automotive dynamic platform model, ground speed reporting, and 3m horizontal accuracy by default. Changes to this configuration are persistent between device restarts.

Example Packet

Send from the client to configure Airplane <4g mode with 3D speed reporting and 2.0 meters of minimum desired horizontal accuracy.

```
B5 62 FF 27 03 00 08 01 02 34 0E
```

Decoding:

Offset	Contents	Field	Decoded
0	`B5 62`	Header	
2	`FF 27`	Message Class and ID	GNSS Config
4	`03 00`	Payload Length	3 bytes
6	`08`	Platform	Airborn
7	`01`	3-d speed reporting	Enable
8	`02`	Minimum accuracy	2.0m
9	`34 0E`	Checksum	

RaceBox Mini S and Micro Standalone Recording

RaceBox Mini S and RaceBox Micro support standalone recording. They can store the sensors' data in the internal memory without a connected BLE client.
Sending any standalone-recording-related commands to a RaceBox Mini (not Mini S or Micro) will result in an error response or no response at all. This section describes commands supported by RaceBox Mini S and Micro only.

Memory Security

The RaceBox Mini S and Micro implement protocol-level protection against unauthorized data recording, access, and erasure. Anyone can connect to the device but if memory security is enabled

all commands related to the standalone recording feature will require unlocking with a preset security code. The feature is not enabled by default. Third-party applications are required to support unlocking if they need to operate with its internal storage. See the Unlock Memory message (0xFF 0x30)

Standalone Recording Status

Message class: 0xFF
Message ID: 0x22
Payload size: 0-12 bytes

To read the standalone recording status send the message with an empty (zero bytes long) payload. The device will reply with a message of the same class and ID with a 12-byte payload with the following structure:

Offset	Size	Type	Description
0	1 byte	Flag	Recording state
1	1 byte	Byte	Memory level
2	1 byte	Bitmask	Memory security status
3	1 byte		Reserved for future use
4	4 byte	UInt32	Stored data messages
8	4 byte	UInt32	Device memory size in data messages

Fields Description

- Recording state - set to a non-zero value if the device is currently recording data.
- Memory level - second byte indicates the memory level in percentage (0..100).
  Note that a value of zero indicates storage is empty and a value of 100% means it is full.
- Security Status - a bitmask with information about the memory security state

BIt 0	1 = Memory security enabled
Bit 1	1 = Memory is unlocked

- Stored data messages and device memory size provide finer details about the used and available storage space, compared to the Memory Level field.

Example Packets

Send from client to the device to request status, including header and checksum in hexadecimal form:

```
B5 62 FF 22 00 00 21 62
```

Returned from the device in response, including header and checksum in hexadecimal form:

```
B5 62 FF 22 0C 00 00 22 01 00 65 06 01 00 00 00 03 00 BF 74
```

Decoding:

Offset	Contents	Field	Decoded
0	B5 62	Header	
2	FF 22	Message class and ID	Recording Config / Status
4	0C 00	Payload length	12 bytes
6	00	Flag	Not Recording
7	22	Memory level	34%
8	01	Memory security	Enabled, Locked
9	00	Reserved	
10	65 06 01 00	Stored data messages	0x10665 = 67173 messages
14	00 00 03 00	Device memory size	0x30000 = 196608 messages
18	BF 74	Checksum	

Standalone Recording Configuration

Message class: 0xFF
Message ID: 0x25
Payload size: 0-12 bytes

To enable/disable standalone recording send the message with 12 bytes payload with the following information:

Offset	Size	Type	Description
0	1 byte	Flag	Enable
1	1 byte	Byte	Data Rate
2	1 byte	Flags	Filters and features flags
3	1 byte	Byte	Reserved for future use
4	2 bytes	UInt16	Stationary filter speed threshold
6	2 byte	UInt16	Stationary filter detection interval

8	2 byte	UInt16	No-Fix filter detection interval
10	2 bytes	UInt16	Auto-Shutdown detection interval

The device will reply to the command with ACK (0xFF 0x02) or NACK (0xFF 0x03) packet to confirm or reject the command. A NACK (0xFF 0x03) will be sent in the following conditions:

- The device does not support standalone recording
- Memory is currently protected
- Configuration is not valid
- There is not enough free memory to start recording (only after FW 3.0)

Send the same message with an empty (zero bytes payload) to read the configuration. The reply payload will be the same 12 bytes as above.

If the Enable byte is set to zero to disable Standalone Recording the device will ignore the rest of the configuration data.

Fields Description

- Enable - set to 1 or 0 to enable/disable standalone recording respectively.
  Note that when you set it to 1 the device will immediately enter recording mode. Live data messages (0xFF 0x01) rate will change according to the value in the Data Rate field.
  The device will keep recording until it receives a command to stop (Enable byte set to zero). In this case, the rest of the configuration is ignored. Clients can safely disconnect and it will keep running.
  You can send repeated Enable packets to change the configuration on the fly. However, beware that each enable command may cause a 1.5-second gap in the data while the internal systems are reconfigured.
- Data Rate - one of the following values:
  - 0: record at 25Hz
  - 1: record at 10Hz
  - 2: record at 5Hz
  - 3: record at 1Hz
  - 4: record at 20Hz (supported on firmware 3.3 and later)

  Note that if a device is configured to <25Hz data rate and you connect to it you will receive the data at the configured frequency until you disable standalone recording. Then it will reset its rate to the default 25Hz
- Filters and features flags - a bitmask to enable the various filters and features (see below for filter details)

BIt 0	1 = Wait for GNSS Location Fix before recording
Bit 1	1 = Enable Stationary filter
Bit 2	1 = Enable No-Fix filter

Bit 3	1 = Enable the Auto-Shutdown feature
Bit 4	1 = Wait for data before shutting down

- Stationary filter speed threshold (in mm/s) and Stationary filter detection interval (in seconds) configure the Stationary Filter (see below)
- No-Fix filter detection interval (in seconds) configures the No-Fix filter (see below)
- Auto-Shutdown detection interval (in seconds) configures the Auto-Shutdown feature (see below)

Recording flags, filters, and features

Wait for GNSS Location Fix

If bit 0 of the flags is set the device will not begin recording until a GNSS location fix is obtained.

Stationary Filter

Enable the stationary filter to avoid recording unnecessary data and save storage space. If the device detects consecutive records below the specified speed (expressed in millimeters per second) for the specified time interval (expressed in seconds) it will stop saving the data. When it detects movement again (even a single record above the specified speed) it will begin recording again.
This filter only activates if there is a valid GNSS location fix.

No-Fix Filter

Enable the No-Fix filter to avoid recording unnecessary data and save storage space. If the device loses the GNSS location fix for longer than the specified time interval (in seconds) it will stop recording new data. If the fix is restored it will begin recording again.

Auto-Shutdown

Enable the Auto-Shutdown feature to preserve battery life. If the device does not record new data for longer than the specified time interval (in seconds) and no BLE client is connected it will completely power off and no new data will be recorded. This option works only if the Stationary Filter, No-Fix Filter, or Wait for GNSS Fix are enabled because otherwise, it will keep recording all the time.
If there is a BLE client connected to the device the Auto-Shutdown timer will not be started until it is disconnected.
If “Wait for data before shutting down” in the Flags is set the Auto-Shutdown timer will not be started unless at least one data record has been saved in the memory since the recording was enabled. If it is not set the device will start its auto-shutdown timer as soon as you disconnect and if it is stationary or doesn’t have a GNSS fix it may shut down before recording anything.

Recommended setup

For general use, we recommend enabling all the filters with stationary filter configuration of 5kph for 30 seconds, No-Fix interval for 30 seconds, and auto-shutdown interval of 5 minutes. If you plan on tracking standing starts the Stationary Filter should be disabled.

RaceBox Micro Recording Start/Stop Button

RaceBox Micro persists settings between device restarts. The user can re-enable recording without a connected BLE client. It will start with the last used recording configuration.

Example Packets

Send from the client, including header and checksum in hexadecimal form with the following contents:

- Enable standalone recording
- Stationary filter enabled: if speed is below 1389 mm/s (about 5 kph) for 30 seconds - stop recording
- No-Fix filter enabled - if the GNSS location fix is lost for 30 seconds - stop recording
- Auto-Shutdown enabled - if no data is recorded (because the device is stationary or the GNSS fix is not recovered) - turn off and wait for a BLE client to reconnect, even if no data has been recorded
- Wait for GNSS location fix before recording begins

```
B5 62 FF 25 0C 00 01 00 1F 00 6D 05 1E 00 1E 00 2C 01 2B 15
```

Decoding:

Offset	Contents	Field	Decoded
0	B5 62	Header	
2	FF 25	Message Class and ID	Recording Config/Status
4	0C 00	Payload Length	12 bytes
6	01	Enable	Enable Recording
7	00	Data Rate	25 Hz
8	1F	Flags mask	Wait for GNSS fix Enable Stationary filter Enable No-Fix filter Enable Auto-shutdown Do not shut down without data
9	00	Reserved	
10	6D 05	Speed threshold	1389 mm/s ~= 5km/h
12	1E 00	Stationary timeout	30 seconds
14	1E 00	No-fix timeout	30 seconds
16	2C 01	Power-off timeout	300 seconds
18	2B 15	Checksum	

Recorded Data Download

Message class: 0xFF
Message ID: 0x23
Payload size: 0, 1, or 4 bytes

To begin downloading the recorded data send the 0xFF 0x23 message with an empty payload (zero bytes length). If the command is accepted the device will reply with the same message but with a 4-byte payload.

Offset	Size	Type	Description
0	4 bytes	UInt32	Expected max history messages

The payload contains the maximum number of stored data records that will be sent to the client. Due to some internal memory alignment and the lack of sophisticated firmware to handle recording pauses, there can be fewer messages than the indicated value. The indicated value is very close and can be used to present download progress or to reserve memory space.
Once the reply is sent the device will begin transmitting its memory contents as History Data messages (0xFF 0x21) and Standalone Recording State Change Messages (0xFF 0x26). Once all data is sent the device will send an ACK (0xFF 0x02) packet with 0xFF 0x23 payload.
To free up bandwidth the device will disable transmission of live data messages (0xFF 0x01) while it is downloading its memory contents.
To cancel the download operation send the same 0xFF 0x23 message with a single-byte payload (doesn't matter what it contains). The device will flush its remaining buffers and finish the operation with an ACK (0xFF 0x02) message. That means you will continue receiving History Data Messages for a few seconds after the cancellation.
As soon as the download is completed (or canceled) the device will resume sending live data messages (0xFF 0x01).
The device will reply with a NACK (0xFF 0x03) message if:

- The device does not support standalone recording
- Memory is currently locked and you need to send the security code first
- Another operation (storage dump or storage erase) is in progress

Note that the data download may take a while on some smartphones. The device can hold up to 196608 records (16.5MB download). Download speed depends on the smartphone, application implementation, and signal strength. It can reach 100KB/s, but in most cases it is about 60KB/s

Example packet to begin download

Send to the device to begin data download, including header and checksum in hexadecimal format:

```
B5 62 FF 23 00 00 22 65
```

Example reply from the device

Sent from the device to indicate start for download with 780 data messages to be sent, including header and checksum in hexadecimal format:

```
B5 62 FF 23 04 00 0C 03 00 00 35 3E
```

Example packet to cancel a download

Send to the device to stop data download, including header and checksum in hexadecimal format:

```
B5 62 FF 23 01 00 FF 22 89
```

History Data Message

Message class: 0xFF
Message ID: 0x21
Payload size: 80 bytes

Once a data download is started the device will begin dumping its memory contents as 0xFF 0x21 messages. Each of them represents a single data record. The payload format is identical to the 0xFF 0x01 Data message.

To optimize the BLE bandwidth the device will always try to send as much data as possible in a BLE notification packet (usually 244 bytes). As a result, notifications will often contain more than one history data packet and part of others. Implementing a byte FIFO buffer where data from the notifications is temporarily stored upon reception is necessary.

Standalone Recording State Change Message

Message class: 0xFF
Message ID: 0x26
Payload size: 12 bytes

Changes to the recording state (start, stop, and pause) are saved in the storage and later transmitted as Standalone Recording State Change Messages. They allow you to identify separate data sets in the memory. The payload contains the new status and information about the current filter settings. For more information about Standalone Recording Filters see the Standalone Recording Configuration message description.

The device will always save a “State change” message (0xFF 0x26) when you start and when you stop recording or when it stops on its own, due to the “Auto Shutdown” feature. It also saves a “pause” message to indicate either the No-Fix Filter or the Stationary Filter kick-in. When resuming after a pause there will not be another “start” message.

If you stay connected after you enable recording you will receive these messages in real-time as well. One notable difference is that transitions from paused to recording state are not saved in memory but reported to connected clients. The reason is that they aren't needed in the recorded stream - the next data message saved after the pause indicates the recording has resumed, but there is no such indication in the live transmission.

Payload:

Offset	Size	Type	Description
0	1 byte	Byte	Standalone Recording State
1	1 byte	Byte	Reserved
2	1 byte	Byte	Data Rate
3	1 byte	Flags	Filters and features flags
4	2 bytes	UInt16	Stationary filter speed threshold
6	2 byte	UInt16	Stationary filter detection interval
8	2 byte	UInt16	No-Fix filter detection interval
10	2 bytes	UInt16	Auto-Shutdown detection interval

Fields description

- Standalone Recording State indicates the new state the device is in. Can have one of the following values:

0	Standalone recording is now disabled
1	Standalone recording is now running
2	Standalone recording is now paused

- The rest of the fields are described in depth in the Standalone Recording Configuration Message and contain the current settings

Example Packets

```
B5 62 FF 26 0C 00 01 00 00 1F 6D 05 0A 00 0A 00 0A 00 E1 F8
```

Decoding:

Offset	Contents	Field	Decoded
0	`B5 62`	Header	
2	`FF 26`	Message Class and ID	Recording State Change

4	0C 00	Payload Length	12 bytes
6	01	New State	1 - Start recording
7	00	Reserved	
8	00	Data Rate	25Hz
9	1F	Flags	Wait for GNSS fix Enable Stationary filter Enable No-Fix filter Enable Auto-shutdown Do not shut down without data
10	6D 05	Stationary speed	1389 mm/s
12	0A 00	Stationary interval	10 seconds
14	0A 00	No-Fix interval	10 seconds
16	0A 00	Auto-Shutdown interval	10 seconds
18	E1 F8	Checksum	

Recorded Data Erase

Message class: 0xFF
Message ID: 0x24
Payload size: 0-1 bytes

To erase (empty) the device's memory contents send this message with an empty payload (zero bytes length). The device will start the erase process immediately. As it executes the request it will send progress notifications with 0xFF 0x24 messages containing one-byte payload - erase progress in percentage (0-100). When it completes the erase it will send an ACK (0xFF 0x02).
The device will reply with a NACK (0xFF 0x03) if it can't start the erase. Possible reasons include:

- The device does not support standalone recording
- Memory is currently locked and you need to send the security code
- Another operation (storage dump or storage erase) is in progress

During the erase live data messages are silenced (0xFF 0x01)
The memory is erased top-to-bottom. That allows cancellation of the process. Send the same 0xFF 0x24 packet with a single byte payload (value does not matter) to interrupt it.
The memory is erased in chunks (64K). Each chunk is checked if it is empty or not and erased if necessary. After each one, the progress is reported. As a result, the progress is not reported in regular intervals but often jumps to some high number and then slowly starts incrementing. Note that full memory erasing may take a couple of minutes.

Example packet to begin erasing

```
B5 62 FF 24 00 00 23 68
```

Example of an erase notification packet

```
B5 62 FF 24 01 00 3B 5F C9
```

The packet indicates erase progress 59% (0x3B)

Example of an erase cancel packet

Send to the device to abort erasing. The example is in hexadecimal form and contains headers and a checksum.

```
B5 62 FF 24 01 00 FF 23 8D
```

Unlock Memory

Message class: 0xFF
Message ID: 0x30
Payload size: 4 bytes

Send this command to a RaceBox Mini S or Micro that has memory protection enabled with the following payload:

Offset	Size	Type	Description
0	4 bytes	UInt32	Security Code

The device will reply with ACK (0xFF 0x02) or NACK (0xFF 0x03) to indicate if the operation was successful or not respectively. The device may disconnect and remain unavailable for connection if several consecutive unsuccessful attempts to unlock the memory are made.
If the device was already unlocked or not secured in the first place the command has no effect and will return ACK even if the code is wrong.
The memory lock state is reset on every new connection. Check the status and unlock it every time you connect to a device.

Example request to unlock

Send to the device to unlock the memory with example security code 123456, including header and checksum in hexadecimal format:

```
B5 62 FF 30 04 00 40 E2 01 00 56 08
```

Decoding:

Offset	Contents	Field	Decoded
0	`B5 62`	Header	

2	FF 30	Message Class and ID	Memory unlock
4	04 00	Payload Length	4 bytes
6	40 E2 01 00	Security code	123456 (0x1E240)
10	56 08	Checksum	

ACK and NACK Reply Messages

Message class: 0xFF
Message ID: 0x02 for ACK, 0x03 for NACK
Payload size: 2 bytes

These messages are sent by the device in response to various commands in the 0xFF class. The payload is two bytes, containing the message class and ID to which this reply relates to.

Example packets

Reply to 0xFF 0x30 Unlock memory command to confirm operation is completed, including message header and checksum in hexadecimal format

```
B5 62 FF 02 02 00 FF 30 32 3A
```

Decoding:

Offset	Contents	Field	Decoded
0	B5 62	Header	
2	FF 02	Message Class and ID	Operation ACK
4	02 00	Payload Length	2 bytes
6	FF 30	Acknowledged operation	Memory Unlock
8	32 3A	Checksum	

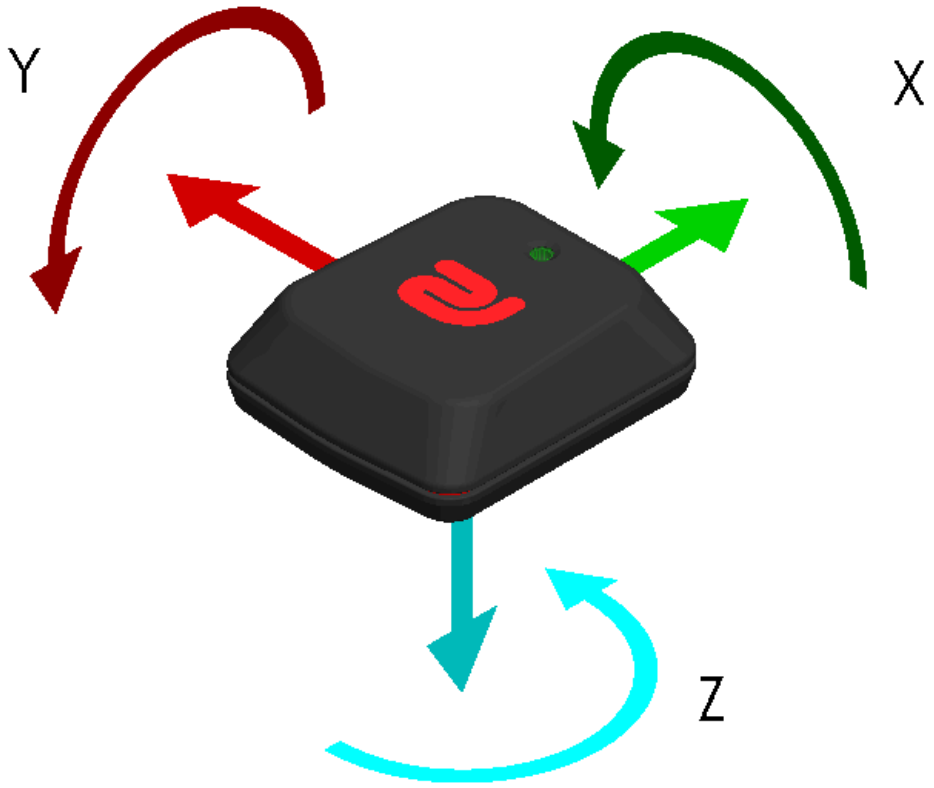
NMEA Standard

Firmware 3.3 and later expose live information from the internal GNSS receiver over a separate BLE characteristic in NMEA 0183 version 4.11 standard. For full details please refer to http://www.nmea.org/.

The provided NMEA sentences are RMC, GNS, GGA, and GSA. The output rate is up to 25Hz. If standalone recording is enabled at a lower data rate the NMEA output will also be reduced. Only live data is sent - there is no support for downloading standalone recordings in NMEA format.

For compatibility reasons, each NMEA message is sent as exactly one BLE notification. Connected clients must support BLE Packet Length Extension and configure PDU size to at least 82 bytes or most packets will be dropped.

Accelerometer and Gyroscope Axes Orientation



Arrow directions indicate positive data value.

Motion Data Handling Recommendations

The device runs its internal accelerometer and gyroscope at 1kHz and averages the readings to produce the 25Hz output. Although this reduces the noise a client may still implement further filtering,

zero-offset calibration, and orientation adjustments based on the application's physical models and needs.

Battery and Input Voltage Level Handling Recommendations

The Mini and Mini S firmware measures the battery voltage every 10 seconds, compensates for load/charging current, and maps that value to a range of 0% to 100%. That approach, while simple and effective, has some caveats in interpreting the resulting value. Most notably it will not always increase/decrease linearly and the value may shift up/down when a charger is plugged/unplugged. Fully charged and fully discharged levels however should be fairly accurate.
The RaceBox Micro does not have a battery. Instead, it reports the input voltage level with <5% accuracy.

Advanced

Internally the device acts almost like a direct pass-through between the GNSS receiver's UART and the client. Most packets sent to the device are directly sent to the internal GNSS receiver and most packets from the receiver are sent to the connected BLE client. The client can reconfigure some receiver settings, send aiding data, or subscribe for additional messages. There are some caveats:

- Packet length cannot be more than 256 bytes due to internal buffer limitation
- Reconfiguring the ports and protocol configurations will likely render the device inoperable.
- The firmware configures the receiver at 25Hz and enables NAV-PVT messages which it captures and overrides with the custom RaceBox Mini data messages.

On RaceBox Mini S the GNSS receiver has battery-backed RAM to improve startup time. On RaceBox Micro there is an optional backup battery. Keep that in mind when sending configuration commands - changes may persist between restarts. However, that may render the device incompatible with other software or completely inoperable. We do not recommend sending configuration changes and will not do warranty replacements or repair of devices damaged by misconfiguration.

For a better user experience, you can improve time-to-first-fix by sending aiding data. You can use U-Blox Assist Now Online to download the relevant commands and pass them on to the device directly.

A full description of the U-Blox UBX packet format and supported commands can be found on the U-Blox website in the Interface Description section.

Document Revisions History

- Revision 1, published 30 Dec 2021

- Revision 2, published 10 January 2022 - added sample RaceBox Mini Data Packet and its decoding
- Revision 3, published 25 January 2022 - fixed incorrect indexing in the pseudocode of the algorithm for packet checksum, added description of the iTOW field; fixed incorrect byte offset of Battery Status and Lat/Lng flags in the data packet description
- Revision 4, published 8 February 2022 - fixed Altitude description - data from the device is in millimeters, not centimeters. Added diagram showing the accelerometer and gyroscope orientations
- Revision 5, published 24 August 2022 - fixed Horizontal Accuracy description - units are millimeters, not centimeters. Also fixed plenty of grammar and spelling errors.
- Revision 6, published 27 March 2023 - added RaceBox Mini S protocol documentation
- Revision 7, NOT published 15 June 2023 - spelling and grammar fixes
- Revision 8, published 16 April 2024 - added details about RaceBox Micro, clarified disabling of standalone recording and auto-shutdown filter in standalone recording mode
- Revision 9, published 4 August 2026 - fixed several typos