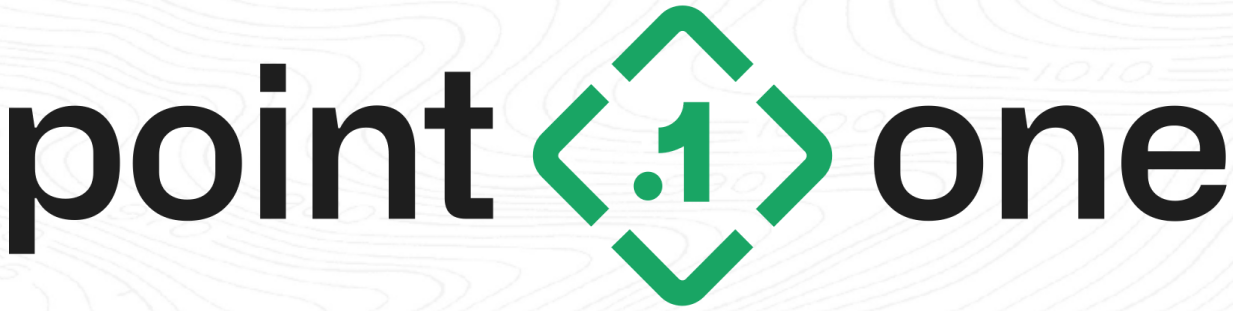


FusionEngine Message Specification

Version 0.22 (2025-3-17)

Compatible with FusionEngine Protocol version 1.24.0

Table Of Contents

1. Overview

This document describes the messages used by Point One's FusionEngine software.

Point One FusionEngine messages are used for both input and output. The messages are a highly compact binary format that contain detailed information about the PVT solution, attitude solution, and other important receiver and navigation statuses. The messages are also used for configuration of the target device, including enabling and disabling other protocols, streams, etc.

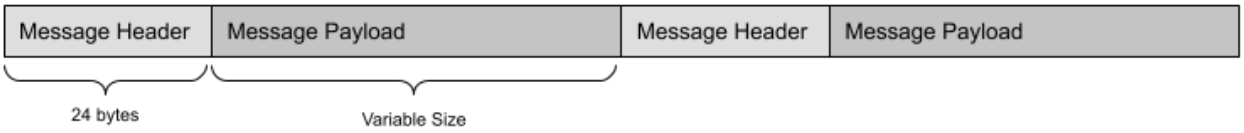
Open-source examples of this protocol are available in the repository linked below. While this manual contains the full descriptions for each message, it is highly recommended to use the existing library whenever possible as it has been thoroughly tested across Point One's products.

The Point One FusionEngine Protocol is available as
an open-source library for C++ and Python:
https://github.com/PointOneNav/fusion-engine-client

Source code documentation is available at:
http://docs.pointonenav.com/fusion-engine

2. Message Structure

All FusionEngine messages consist of a Message Header and Message Payload. Messages are aligned to 4-byte boundaries, so the size of a message will *always* be a multiple of 4 bytes. This means that messages will contain padding bytes if necessary.



2.1 Data Types

Data Type	Size in bytes	Description
`bool`	1	Boolean Value (0 or 1)
`u8`	1	Unsigned Integer
`u16`	2	Unsigned Integer
`u32`	4	Unsigned Integer
`u8[N]`	N	Reserved/Padding Bytes
`i8`	1	Signed Integer
`i16`	2	Signed Integer
`i32`	4	Signed Integer
`i64`	8	Signed Integer
`f32`	4	IEEE Single-Precision Floating Point
`f64`	8	IEEE Double-Precision Floating Point
`str`	*Variable*	Character String (Not Null-Terminated)

2.2 Byte Ordering

All FusionEngine messages are constructed using little endian (Intel) encoding.

2.3 Message Header

The message header is 24 bytes in length and contains synchronization bytes and information required to decode or encode the Message Payload.

Field	Data Type	Description
Sync Byte 0	`u8`	Always 0x2E (ASCII ".")
Sync Byte 1	`u8`	Always 0x31 (ASCII "1")
Reserved	`u8[2]`	*Reserved for future use.*
CRC	`u32`	The 32-bit CRC of all bytes from and including the protocol version field to the last byte in the message, including the message payload. This uses the standard CRC-32 generator polynomial in reversed order (0xEDB88320).
Protocol Version	`u8`	The version of the FusionEngine Protocol.
Message Version	`u8`	The version of this message. Messages of the same type will always be backwards compatible by using reserved fields in previous versions or extending the message length.
Message Type	`u16`	Uniquely identifies each message type. The combination of Version and Type should be used to know how to decode the payload.
Sequence Number	`u32`	A sequence number that is incremented with each message.
Payload Size	`u32`	Size of the payload to follow in bytes.
Source Identifier	`u32`	Identifies the source of the message when applicable. This definition can change depending on the message type.

2.4 CRC Calculation Details

Below is an abbreviated example of the CRC algorithm. Note that some hardware may have dedicated accelerators for this algorithm but that is out of scope of this document. See `crc.cc` in the open-source library for a complete reference.

Generation of the *CRC Table*

```
// This table can be pre-computed once and reused for all messages.
// It does not depend on data content.
const uint32_t polynomial = 0xEDB88320;
for (uint32_t i = 0; i < 256; i++) {
  uint32_t c = i;
  for (size_t j = 0; j < 8; j++) {
    if (c & 1) {
      c = polynomial ^ (c >> 1);
    } else {
      c >>= 1;
    }
  }
  crc_table[i] = c;
}
```

Calculating the CRC on a *buffer* of known *length*

```
uint32_t initial_value = 0;
uint32_t c = initial_value ^ 0xFFFFFFFF;
for (size_t i = 0; i < length; ++i) {
  c = crc_table[(c ^ buffer[i]) & 0xFF] ^ (c >> 8);
}
return c ^ 0xFFFFFFFF;
```

2.5 Timing

All Point One devices generate a common time base called Point One time (P1 time). P1 time is used to coordinate all sensor measurement and solution data within the system.

P1 time may be driven by a microcontroller (MCU), or a local monotonic CPU clock provided by the operating system. P1 time is synthetic and may be adjusted dynamically based on internal clock models as the system operates. Users should not assume that P1 time is equal to the source clock (MCU, OS, or POSIX time), or to GPS time.

In addition to P1 time, some messages may optionally include additional timestamps where applicable, including GPS system time, POSIX time, and platform OS time. See Section 2.6.2 for more details.

2.6 Common Data Structure/Enumeration Definitions

This section defines some of the common data structures and enumerations that are used within multiple message types.

2.6.1 Timestamp

This structure may be used to convey P1 time values (referenced to the start of the device), GPS times (referenced to January 6, 1980), POSIX times (referenced to January 1, 1970), or embedded device system timestamps (platform-specific). The type of time will be specified by the SystemTimeSource (see Section 2.6.2).

`Timestamp` (8B):

Field	Data Type	Description
Seconds	`u32`	The number of integer seconds since the reference epoch. Set to 0xFFFFFFFF if invalid.
Fraction	`u32`	The fractional part of the timestamp in nanoseconds. Set to 0xFFFFFFFF if invalid.

2.6.2 SystemTimeSource

Represents the time base used for a message timestamp. When a measurement is supplied externally, it may either contain a timestamp generated by the sender or it can be timestamped by FusionEngine when the packet arrives.

`SystemTimeSource` (u8):

Type	Value	Description
Invalid	`0x0`	Timestamp not valid. Message will be timestamped on arrival.
P1 Time	`0x1`	Measurement timestamped in P1 time.
Timestamped on Reception	`0x2`	Message timestamped in device system time at the time of arrival.
Sender System Time	`0x3`	Timestamp was generated from a monotonic clock of an external system.
GPS Time	`0x4`	Message timestamped in GPS time, referenced to 1980/1/6.
`Reserved`	`0x5-0xFF`	*Reserved for future use.*

2.6.3 SensorDataSource

Sensor data may come from a variety of different sources. This value identifies the sensor data source and can be useful for verification purposes when viewing output messages.

`SensorDataSource` (u8):

Type	Value	Description
Unknown	`0x0`	The data source is unknown.
Internal	`0x1`	Sensor data captured internally within the device (embedded IMU, internal GNSS receiver, etc.)
Hardware IO	`0x2`	Sensor data generated via hardware voltage signal measured directly by the device (hardware wheel tick, external event, etc.).
CAN	`0x3`	Sensor data captured from a platform CAN bus.
Serial	`0x4`	Sensor data input over a serial (UART) connection
Network	`0x5`	Sensor data input over a network (Ethernet, Wi-Fi, etc.)
`Reserved`	`0x6-0xFF`	*Reserved for future use.*

2.6.4 MeasurementDetails

All messages containing sensor measurements, either external inputs to a FusionEngine device or "raw" (uncorrected) measurement output messages, begin with a `MeasurementDetails` structure. This structure describes the time of applicability of the sensor data.

Where possible, the user may provide known P1, GPS, or system timestamps indicating the time at which the sensor data applies. If a timestamp cannot be provided, the incoming sensor data will be timestamped by FusionEngine on arrival.

Note that measurements timestamped on arrival are subject to unknown latency due to the capture and transmission of the sensor data, including latency introduced by the data interface. Excessive latency may result in performance degradation.

Once the message is received, the specified measurement time will be converted to P1 time automatically by FusionEngine using its internal clock models. The user should not populate the output P1 time field when sending measurement data to a device. Any value in the P1 time field will be ignored.

`MeasurementDetails (20B):`

Field	Data Type	Description
Measurement Time	`Timestamp (8B)`	The time of applicability of the measurement data. Interpretation dictated by Time Source.
Time Source	`SystemTimeSource (u8)`	The source of the measurement timestamp. See Section 2.6.2.
Sensor Data Source	`SensorDataSource (u8)`	The source of the sensor. See Section 2.6.3
Reserved	`u8[2]`	*Reserved for future use.*
Output P1 Time	`Timestamp (8B)`	Leave invalid when sending measurements to the device. On output, this will be the calculated P1 time corresponding with the measurement timestamp.

2.6.5 GNSS System/Signal Type Constants (`SatelliteType`)

The following enumeration is used to specify GNSS systems (constellations).

`SatelliteType (u8):`

Name	Enum Value (u8)	Bit Mask (u16)	Description
UNKNOWN	0	N/A	GNSS constellation not known.
GPS	1	0x002	
GLONASS	2	0x004	
LEO	3	0x008	*Not currently used.*
GALILEO	4	0x010	
BEIDOU	5	0x020	
QZSS	6	0x040	
MIXED	7	0x080	*Not currently used.*
SBAS	8	0x100	
IRNSS	9	0x200	Also known as NavIC.
Reserved	10-255		*Reserved for future use.*

When sending a SetConfig (13100) message to enable/disable GNSS constellations, use the values in the Bit Mask column. For example, the mask 0x12 enables GPS and Galileo.

`GPSSignalType (u8):`

Name	Enum Value (u8)	Bit Mask (u16)
L1 C/A	0	0x0001
L1 P(Y)	1	0x0002
L1C	2	0x0004
Reserved	3	
L2C	4	0x0010
L2P	5	0x0020
Reserved	6-7	
L5	8	0x0100
Reserved	9-255	

`GLOSignalType (u8):`

Name	Enum Value (u8)	Bit Mask (u16)
L1 C/A	0	0x0001
L1 P	1	0x0002
Reserved	2-3	
L2 C/A	4	0x0010
L2 P	5	0x0020
Reserved	6-15	

GALSignalType (u8):

Name	Enum Value (u8)	Bit Mask (u16)
E1-A	0	0x0001
E1-BC	1	0x0002
Reserved	2-3	
E5b	4	0x0010
Reserved	5-7	
E5a	8	0x0100
Reserved	9-11	
E6-A	12	0x1000
E6-BC	13	0x2000
Reserved	14-15	

BDSSignalType (u8):

Name	Enum Value (u8)	Bit Mask (u16)
B1I	0	0x0001
B1C	1	0x0002
Reserved	2-3	
B2I	4	0x0010
B2b	5	0x0020
Reserved	6-7	
B2a	8	0x0100
Reserved	9-11	

B3	12	0x1000
Reserved	13-15	

2.6.6 GNSS Frequency Band Constants (`FrequencyBand`)

The following enumeration is used to specify GNSS systems frequency bands. Each frequency band spans a range of GNSS as shown below.

`FrequencyBand (u8):`

Name	Value	Bit Mask	Description
UNKNOWN	0x0		Frequency band not known.
L1	0x1	0x02	1561.098 MHz (B1) -> 1602.0 (G1) Includes: • GPS/QZSS L1 (1575.42 MHz) • Galileo E1 (1575.42 MHz) • BeiDou B1I (1561.098 MHz) • BeiDou B1C (1575.42 MHz) • GLONASS G1 (1602.0 MHz)
L2	0x2	0x04	1202.025 MHz (G3) -> 1248.06 (G2) Includes: • GPS L2 (1227.6 MHz) • Galileo E5b (1207.14 MHz) • BeiDou B2I (1207.14 MHz) • GLONASS G2 (1246.0 MHz) • GLONASS G3 (1202.25 MHz)
Reserved	0x3–x4		*Reserved for future use.*
L5	0x5	0x20	1176.45 MHz (L5/E5a/B2a) Includes: • GPS/QZSS L5 (1176.45 MHz) • Galileo E5a (1176.45 MHz) • BeiDou B2a (1176.45 MHz) • IRNSS L5 (1176.45 MHz)
L6	0x6	0x40	1262.52 MHz (B3) -> 1278.75 (L6/E6) Includes: • Galileo E6 (1278.75 MHz) • BeiDou B3 (1268.52 MHz) • QZSS L6 (1278.75 MHz)
Reserved	0xA–0xFF		*Reserved for future use.*

When sending a SetConfig (13100) message to enable/disable GNSS frequencies, use the values in the Bit Mask column. For example, the mask 0x22 enables the L1 and L5 bands.

2.6.7 Transmission Gear Type (`GearType`)

The following enumeration defines the “gear” in a vehicle transmission.

`GearType` (u8):

Stage	Value	Description
Unknown	`0x0`	Transmission gear not known, or gear does not map to one of the values defined below.
Forward	`0x1`	Vehicle in a forward gear.
Reverse	`0x2`	Vehicle in a reverse gear.
Park	`0x3`	Vehicle is parked.
Neutral	`0x4`	Vehicle is in neutral.
Reserved	`0x5-0xFF`	*Reserved for future use.*

2.6.8 I/O Interface ID (`InterfaceID`)

Describes an individual I/O interface (e.g., serial UART 1, TCP socket 3).

`InterfaceID (u32):`

Field	Data Type	Description
Transport Type	`TransportType (u8)`	The type of the IO interface.
Index	`u8`	The ID of the IO interface. For UARTs, typically UART1 is index 1.
Reserved	`u8[2]`	*Reserved for future use.*

`TransportType (u8):`

Describes the type of transport used by an I/O interface.

Name	Value	Description
Invalid	`0x0`	Not used
Serial	`0x1`	A serial data interface (e.g. an RS232 connection).
File	`0x2`	An interface that writes to or reads from a file.
TCP	`0x3`	A TCP client or server.
Reserved	`0x4`	*Reserved for future use.*
UDP	`0x5`	A UDP client or server.
Reserved	`0x6`	*Reserved for future use.*
WebSocket	`0x7`	A WebSocket client or server.
UNIX Domain Socket	`0x8`	A UNIX domain socket client or server.
Current	`0xFE`	The interface on which the command was received (used to query transport or message rate settings for the interface currently being used).
All	`0xFF`	Wildcard to get/set ALL interfaces. *Not supported for all commands.*

2.6.9 Stored Data Type (`DataType`)

Identifies the type of data stored on the device.

`DataType (u8):`

Data Type	Data Type	Description
CALIBRATION_STATE	0x0	Active configuration settings currently in use by the device.
CRASH_LOG	0x1	Diagnostic information reported in the case of an unexpected crash.
FILTER_STATE	0x2	Filter state parameters.
USER_CONFIG	0x3	User configuration parameters.
INVALID	0xFF	Invalid value.

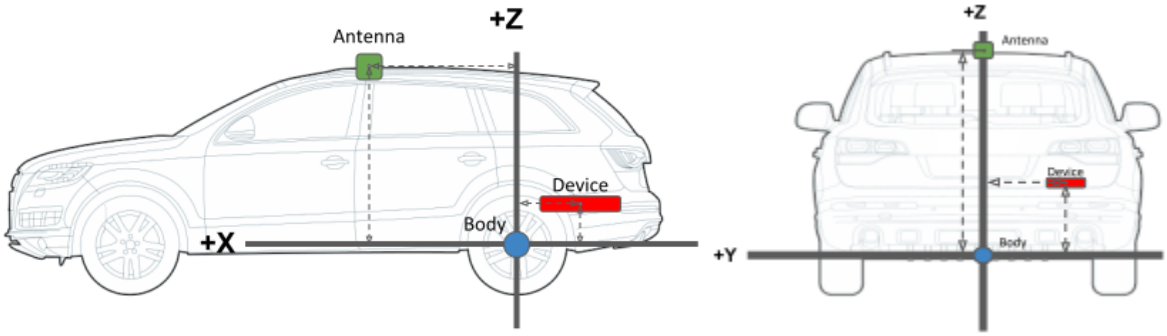
2.6.10 Data Version Number (`DataVersion`)

Defines the version of the data structure definition currently in use by the device.

`DataVersion (u32):`

Field	Data Type	Description
Reserved	`u8`	Must be set to 0xFF for backwards compatibility.
Major Version	`u8`	Major version number. Set to 0xFF if invalid.
Minor Version	`u16`	Minor version number. Set to 0xFFFF if invalid.

2.7 Vehicle Body Coordinate Frame And Attitude Angles



The vehicle body axes used in all FusionEngine messages are defined as:

- +X forward
- +Y left
- +Z up

The attitude angles used in all messages are defined with respect to the local ENU (East, North, Up) reference frame as follows:

- Yaw – A right-hand (counter-clockwise) rotation around the Z axis, starting in the east. For example, a yaw of +90° indicates the vehicle X axis is pointing to the north.
    - Heading with respect to true north can be computed as: heading = 90 - yaw
- Pitch – A right-hand rotation around the Y axis. A positive pitch angle points the nose of the vehicle down relative to the horizon, while a negative pitch points the vehicle up.
- Roll – A right-hand rotation around the X axis. A positive roll angle indicates a roll to the right when looking towards the front of the vehicle, a negative roll indicates a roll to the left.

These angles are intrinsic Euler-321 angles. The order in which the rotations are applied is important. The correct order is: yaw, pitch, then roll.

2.8 NMEA-0183 Message Types

The following table lists the 16-bit (u16) message IDs assigned to each supported NMEA-0183 message, used to configure output interface settings. All other values are reserved for future use.

See also 4 Proprietary NMEA Sentence Definitions.

NMEA Message Type	Message ID
Invalid	0
Standard NMEA-0183 Messages	
GGA	1
GLL	2
GSA	3
GSV	4
RMC	5
VTG	6
ZDA	7
Point One Proprietary Messages	
P1CALSTATUS	1000
P1MSG	1001
Quectel Proprietary Messages	
PQTMVERNO	1200
PQTMVER	1201
PQTMGNSS	1202
PQTMVERNO_SUB	1203
PQTMVER_SUB	1204
PQTMTXT	1205

3. FusionEngine Message Definitions

3.1 Command/Control Messages (Input)

These are messages sent from the host device to the FusionEngine software. They are primarily used to control the device and to change or query configuration settings.

3.1.1 MessageRequest (13001)

Message ID: 13001

Protocol Version: 2
Message Version: 0

Description: Request transmission of a specific message type. Unless otherwise stated, the response will be generated on the same interface.

On-demand message transmission is not supported for all message types.

Field	Data Type	Description
Message Type	`u16`	The type of the message requested.
Reserved	`u8[2]`	*Reserved for future use.*

Expected Response(s):

Responding Message Type	Responding Message ID
<Requested Message Type>	<Requested Message Type ID>

Note: If the specified message type cannot be requested, a 3.2.1 CommandResponse (13000) will be returned with a *Value Error*.

3.1.2 ResetRequest (13002)

Message ID: 13002

Protocol Version: 2
Message Version: 0

Description: Command the system to perform one or more reset operations.

Field	Data Type	Description
Reset Mask	`u32`	Bit mask indicating the components to be reset.

Reset Mask Bits:

Bit Number	Name	Reset Triggered When Set
0	`RESTART_NAVIGATION_ENGINE`	Restart the navigation engine, but do not clear its position estimate.
1	`RESET_GNSS_CORRECTIONS`	Delete all GNSS corrections information.
2	`RESET_GNSS_TIME`	Delete all GNSS time information.
3-7	Reserved	*Reserved for future use.*
8	`RESET_POSITION_DATA`	Reset the navigation engine's estimate of position, velocity, and orientation.
9	`RESET_EPHEMERIS`	Delete all saved satellite ephemeris.
10	`RESET_FAST_IMU_CORRECTIONS`	Reset bias estimates, and other IMU corrections that are typically estimated quickly.
11	Reserved	*Reserved for future use.*
12	`RESET_NAVIGATION_ENGINE_DATA`	Reset all stored navigation engine data, including position, velocity, and orientation state, as well all IMU corrections (fast and slow) and other training data.
13	`RESET_CALIBRATION_DATA`	Reset the device calibration data. Note: This does not reset any existing navigation engine state. It is recommended that you set `RESET_NAVIGATION_ENGINE_DATA` as well under normal circumstances.
14-19	Reserved	*Reserved for future use.*

20	RESET_CONFIG	Reset all configuration settings back to factory defaults.
21-23	Reserved	*Reserved for future use.*
24	REBOOT_GNSS_MEASUREMENT_ENGINE	Reboot the GNSS measurement engine, in addition to performing any other requested resets. If no other resets are specified, the GNSS receiver will reboot and should perform a hot start.
25	REBOOT_NAVIGATION_PROCESSOR	Reboot the processor on which the navigation software is executed.
26	DIAGNOSTIC_LOG_RESET	Perform a diagnostic log reset to guarantee deterministic performance for data post-processing and diagnostic support.
26-31	Reserved	*Reserved for future use.*

Special Mask Values:

Value	Name	Description
0x00000001	HOT_START	Perform a device hot start. A hot start is typically used to restart the navigation engine in a deterministic state, using previously stored position and time information. The device will begin navigating immediately if possible. To be reset: • The navigation engine Not reset: • All runtime data (GNSS corrections, etc.) • GNSS time information • Position, velocity, orientation state • GNSS ephemeris data • Fast IMU corrections • Training parameters (slowly estimated IMU corrections, temperature compensation, etc.) • Calibration data • User configuration settings • GNSS measurement engine • Navigation processor

0x00000201	WARM_START	Perform a device warm start. During a warm start, the device retains its knowledge of approximate position and time, plus almanac data if available, but resets all ephemeris data. As a result, the device will need to download ephemeris data before continuing to navigate. To be reset: ● The navigation engine ● GNSS ephemeris data Not reset: ● All runtime data (GNSS corrections, etc.) ● GNSS time information ● Position, velocity, orientation state ● Fast IMU corrections ● Training parameters (slowly estimated IMU corrections, temperature compensation, etc.) ● Calibration data ● User configuration settings ● GNSS measurement engine ● Navigation processor

0x00000FFF	COLD_START	Perform a device cold start. A cold start is typically used to reset the device's state estimate in the case of an error that cannot be resolved by a warm start. To be reset: • The navigation engine • All runtime data (GNSS corrections, etc.) • GNSS time information • Position, velocity, orientation state • Fast IMU corrections • GNSS ephemeris data Not reset: • Training parameters (slowly estimated IMU corrections, temperature compensation, etc.) • Calibration data • User configuration settings • GNSS measurement engine • Navigation processor
0x000001FB	POSE_RESET	Resets the kinematic portion of the navigation engine state. Useful for error recovery. To be reset: • The navigation engine • All runtime data (GNSS corrections, etc.) • Position, velocity, orientation state Not reset: • GNSS time information • GNSS ephemeris data • Fast IMU corrections • Training parameters (slowly estimated IMU corrections, temperature compensation, etc.) • Calibration data • User configuration settings • GNSS measurement engine • Navigation processor
0xFFFFFFFF	FACTORY_RESET	Restart mask to set all persistent data, including position/orientation, calibration state, and user configuration, back to factory defaults.

Expected Response(s):

3.2.1 CommandResponse (13000)

Example Usage

Example 1: Perform a device cold start

This will reset the navigation engine including saved position, velocity, and orientation state, as well as the GNSS measurement engine, but will not reset training data, calibration data, or user configuration parameters, by setting the reset mask to 0x00000FFF.

```
2e31 0000 0acf ee8f 0200 ca32 0000 0000
0400 0000 0000 0000 ff0f 0001
```

Example 2: Perform a factory reset

Reset all settings, state, and calibration data to factory defaults by setting the reset mask to 0xFFFFFFFF.

```
2e31 0000 b39e 0bf3 0200 ca32 0000 0000
0400 0000 0000 0000 ffff ffff
```

3.1.3 ShutdownRequest (13005)

Message ID: 13005

Protocol Version: 2
Message Version: 0

Description: Shutdown the device.

Field	Data Type	Description
Flags	`u64`	A bitmask of flags used to control the shutdown process. Note: if the flags field is set to zero, it will be treated as a “Stop Engine” request.
Reserved	`u8[8]`	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
Stop Engine	`0`	`0x1`	Stop navigation engine and flush state to non-volatile storage.
Stop Current Log	`1`	`0x2`	If a log is being generated, end that log.
Reserved	`2-63`		*Reserved for future use.*

Expected Response(s):

3.2.1 CommandResponse (13000)

Example Usage

Example 1: Stop the navigation engine

```
2e31 0000 716f 5414 0200 cd32 0000 0000
1000 0000 0000 0000 0000 0000 0000 0000
0000 0000 0000 0000
```

3.1.4 FaultControl (13006)

Message ID: 13006

Protocol Version: 2
Message Version: 0

Description: Testing message to trigger or clear a specified system fault.

This message is intended for factory testing and development purposes only and should not be used in any production environment.

Field	Data Type	Description
Fault Type	`FaultType (u8)`	The type of fault/control to be performed.
Reserved	`u8[15]`	*Reserved for future use.*
Value Length	`u32`	The length of the value to follow (in bytes).
Value	*Variable*	The parameter value for the specified fault type. The format of the value is listed in the `FaultType` table below.

`FaultType` (u8)*:*

Name	Type	Value Format	Description
Clear All	`0x0`	None	Clear existing faults.
Crash	`0x1`	None	Force the device to crash. In a crash state, the device no longer produces any output on any interfaces, and will stop responding to commands. If the watchdog is enabled, the device will restart automatically after the watchdog timer elapses.
Fatal Error	`0x2`	None	Force the device to exhibit a fatal error. After a fatal error, the device will stop navigating and will no longer produce solution messages on any interfaces. Instead, it will output an `EventNotificationMessage` indicating the fault status. If the watchdog is enabled, the device will restart automatically after the watchdog timer elapses.

COCOM	`0x3`	COCOM Type (3.1.4.1) (u8)	Trigger a COCOM limit. When a COCOM limit is exceeded, the device will stop navigating and will produce Invalid solution messages. COCOM limits may be cleared via ResetRequest, or by sending a COCOM FaultControl message with the `COCOMType` parameter set to `NONE`.
Enable GNSS	`0x4`	`bool`	Enable/disable use of GNSS measurements (intended for dead reckoning performance testing).
Region Blackout	`0x5`	`bool`	Enable/disable a simulated region blackout
Line Test Mode	`0x6`	`bool`	Enable/disable factory test behavior for use with GNSS simulation.
Integrity Status	`0x7`	`u8`	Simulate a specified integrity status failure.
Reserved	`0x7-0xFF`		*Reserved for future use.*

Expected Response(s):

3.2.1 CommandResponse (13000)

Note that the device will crash immediately after receiving a Crash Fault Type request. It will not send a CommandResponse back to the user.

Example Usage

Example 1: Cause the device to immediately crash.

```
2e31 0000 bde4 777b 0200 ce32 0000 0000
1400 0000 0000 0000 0100 0000 0000 0000
0000 0000 0000 0000 0000 0000
```

Example 2: Trigger an artificial COCOM altitude limit.

```
2e31 0000 0bf7 0344 0200 ce32 0000 0000
1500 0000 0000 0000 0300 0000 0000 0000
0000 0000 0000 0000 0100 0000 03
```

Note that, in practice, the message sequence number should be modified, consistent with any commands sent previously, and the CRC should be updated to reflect the change.

3.1.4.1 COCOM Type (u8)

Name	Value	Description
NONE	0x0	Clear the current COCOM limit.
ACCELERATION	0x1	Trigger a maximum acceleration limit.
SPEED	0x2	Trigger a maximum speed limit.
ALTITUDE	0x3	Trigger a maximum altitude limit.
Reserved	0x4-0xFF	*Reserved for future use.*

3.1.5 StartupRequest (13008)

Message ID: 13008

Protocol Version: 2
Message Version: 0

Description: Startup the device.

Field	Data Type	Description
Flags	`u64`	A bitmask of flags used to control the startup process. Note: For backwards-compatibility, if set to 0x0, the navigation engine will be started.
Reserved	`u8[8]`	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
Start Engine	`0`	`0x1`	Start navigation engine if not currently running.
Start New Log	`1`	`0x2`	If a log is not being generated, start a new log. If a log is active, end it and immediately start a new log.
Reserved	`2-63`		*Reserved for future use.*

Expected Response(s):

3.2.1 CommandResponse (13000)

3.1.6 SetConfigMessage (13100)

Message ID: 13100

Protocol Version: 2
Message Version: 0

Description: Set the value of a user configuration parameter.

Field	Data Type	Description
Parameter Type	`ConfigType (u16)`	The parameter to be configured.
Flag	`u8`	Bitmask of additional flags to modify the command.
Reserved	`u8`	*Reserved for future use.*
Value Length	`u32`	The length of the value to follow (in bytes).
Value	*Variable*	The new value to use for the specified parameter. The format of the value is listed in the `ConfigType` table below.

Flag bit definitions:

Flag	Bit	Value	Description
`APPLY_AND_SAVE`	`0`	`0x1`	Immediately save the configuration after applying this setting.
`REVERT_TO_DEFAULT`	`1`	`0x2`	Reset both the active and saved configuration to default values.
`Reserved`	`2-7`		*Reserved for future use.*

`ConfigType (u16)`*:*

Name	Parameter Type	Value Format	Description
Invalid	`0x0`	None	Invalid parameter.
Reserved	`0x1-0xF`		*Reserved for future use.*
Device Lever Arm	`0x10`	Point3f (3.1.6.1)	The location of the device IMU with respect to the vehicle body frame, resolved in the body frame, in meters.
Device Orientation	`0x11`	CoarseOrientation (3.1.6.2)	The orientation of the device IMU with respect to the vehicle body axes.
GNSS Lever Arm	`0x12`	Point3f (3.1.6.1)	The location of the primary GNSS antenna with respect to the vehicle body frame, resolved in the body frame, in meters.

Output Lever Arm	`0x13`	Point3f (3.1.6.1)	The offset of the desired output location with respect to the vehicle body frame, resolved in the body frame, in meters.
Vehicle Details	`0x14`	VehicleDetails (3.1.6.3)	Information about the vehicle including model and dimensions.
Software Wheel Config	`0x15`	WheelConfig (3.1.6.4)	Configuration for software wheel speed/rotation (tick) measurements (FusionEngine messages/CAN data).
Hardware Tick Config	`0x16`	HardwareTickConfig (3.1.6.5)	Configuration capturing hardware wheel tick measurements from an incoming voltage signal.
Heading Bias *(Deprecated)*	`0x17`	HeadingBias (*removed*)	Used to set horizontal (yaw) & vertical (pitch) biases (in degrees) on a dual-antenna heading platform configuration. *Deprecated in version 1.24.0: Use GNSS Aux Lever Arm instead.*
GNSS Aux Lever Arm	`0x18`	Point3f (3.1.6.1)	The location of the secondary GNSS antenna on a dual-antenna platform. Specified with respect to the vehicle body frame, resolved in the body frame, in meters. For dual-antenna systems, the secondary or auxiliary antenna is used to measure vehicle orientation (yaw/heading and pitch). *Added in version 1.24.0.*
Enabled GNSS Systems	`0x32`	`u32`	A bitmask indicating which GNSS systems are enabled (see 2.6.5 GNSS Systems (SatelliteType)).
Enabled GNSS Frequency Bands	`0x33`	`u32`	A bitmask indicating which GNSS frequency bands are enabled (see 2.6.6 GNSS Frequency Bands (FrequencyBand)).

Leap Second Override	0x34	i32	Specify a UTC leap second count override value to use for all UTC time conversions. Note: Setting this value will disable all internal leap second sources and should **only be used for testing.** Set to -1 to disable.
GPS Week Rollover Override	0x35	i32	Specify a GPS legacy week rollover count override to use when converting all legacy 10-bit GPS week numbers. Setting this value will disable all internal week rollover sources and **should only be used for testing.** Set to -1 to disable.
Ionosphere Config	0x36	Ionosphere Delay Model (3.1.6.7)	Configures the active ionosphere model.
Troposphere Config	0x37	Troposphere Delay Model (3.1.6.8)	Configures the active troposphere model.
Interface Config	0xC8	InterfaceConfig (3.1.6.6)	Change a configuration setting for a specified output interface.
UART 1 Baud Rate	0x100	u32	The UART 1 serial baud rate in bits/second. **Deprecated: Use InterfaceConfig.**
UART 2 Baud Rate	0x101	u32	The UART 2 serial baud rate in bits/second. **Deprecated: Use InterfaceConfig.**
UART 1 Diag Enable	0x102	bool	Enable/disable diagnostic output on UART 1. **Deprecated: Use InterfaceConfig.**
UART 2 Diag Enable	0x103	bool	Enable/disable diagnostic output on UART 2. **Deprecated: Use InterfaceConfig.**
Reserved	0x104-0x12B		*Reserved for future use.*
Watchdog Enable	0x12C	bool	Enable/disable watchdog timer to restart automatically after fatal errors.
User Device ID	0x12D	char[32]	A string for identifying a device.
Profiling Mask	0x136	u8	A bitmask indicating which profiling features are enabled. Individual bit definitions vary per device.
Reserved	0x12E-0x3FF		*Reserved for future use.*

L-band Parameters	0x400	LBandConfig (3.1.6.9)	Configuration of L-band Demodulator Parameters. Note: This setting is only available on devices with an L-band receiver.
Reserved	0x401-0xFFFF		*Reserved for future use.*

Expected Response(s):

3.2.1 CommandResponse (13000)

Example Usage

Example 1: Set the output lever arm to [0.6, 0.0, 1.2] meters

```
2e31 0000 8cd8 859a 0200 2c33 0000 0000
1400 0000 0000 0000 1300 0000 0c00 0000
9a99 193f 0000 0000 9a99 993f
```

Example 2: Set the device orientation to point towards the left side of the vehicle and save to persistent storage immediately.

```
2e31 0000 0ac0 a42c 0200 2c33 0000 0000
0c00 0000 0000 0000 1100 0100 0400 0000
0204 0000
```

Example 3: Set the device UART 1 serial baud rate *to 115200 bits/second.*

```
2e31 0000 61ad a75c 0200 2c33 0000 0000
0c00 0000 0000 0000 0001 0100 0400 0000
00c2 0100
```

The command response will be sent at the new baud rate if the command was sent to the device on the affected UART.

Note that in practice, the message sequence number should be modified, consistent with any commands sent previously, and the CRC should be updated to reflect the change.

3.1.6.1 Three-Dimensional Vector (Point3f)

Field	Data Type	Description
X	f32	The X axis offset in meters.
Y	f32	The Y axis offset in meters.
Z	f32	The Z axis offset in meters.

3.1.6.2 Device Coarse Orientation (CoarseOrientation)

A device's orientation is defined by specifying how the +X and +Z axes of its IMU are aligned with the vehicle body axes. The vehicle body is defined with +X facing forward, +Y facing left, and +Z facing up (forward-left-up).

Field	Data Type	Description
X Direction	`u8`	The direction of the +X axis relative to the vehicle body axes.
Z Direction	`u8`	The direction of the +Z axis relative to the vehicle body axes.
Reserved	`u8[2]`	*Reserved for future use.*

`Direction (u8):`

Name	Value	Description
Forward	`0x0`	Aligned with the vehicle +X axis.
Backward	`0x1`	Aligned with the vehicle -X axis.
Left	`0x2`	Aligned with the vehicle +Y axis.
Right	`0x3`	Aligned with the vehicle -Y axis.
Up	`0x4`	Aligned with the vehicle +Z axis.
Down	`0x5`	Aligned with the vehicle -Z axis.
Reserved	`0x6-0xFE`	*Reserved for future use.*
Invalid	`0xFF`	Invalid direction value.

3.1.6.3 Vehicle Details and Physical Dimensions (VehicleDetails)

This payload is used to configure the vehicle's physical dimensions, used when applying differential wheel speed measurements, as well as select the vehicle model from a list of supported vehicles when processing vehicle CAN messages.

Field	Data Type	Description
Model	`VehicleModel (u16)`	The vehicle model if one of the supported vehicles in the list below. Used to configure CAN data support. Set to 0 (unknown) if not one of the currently supported types.
Reserved	`u8[10]`	*Reserved for future use.*
Wheelbase	`f32`	The distance between the front and rear axles in meters.
Front Track Width	`f32`	The distance between the two front wheels in meters.
Rear Track Width	`f32`	The distance between the two rear wheels in meters.

`VehicleModel (u16):`

Value	Description
`0x0`	Unknown vehicle type
`0x1`	Dataspeed CD4 drive-by-wire system
`0x2`	J1939 CAN message standard

0x14	Lexus CT200H
0x28	Kia Sorento
0x29	Kia Sportage
0x3C	Audi Q7
0x3D	Audi A8L
0x50	Tesla Model X
0x51	Tesla Model 3
0x64	Hyundai Elantra
0x78	Peugeot 206
0x8C	MAN TGX
0xA0	Faction
0xA1	Faction V2
0xB4	Lincoln MKZ
0xC8	BMW 7-Series
0xC9	BMW-Motorrad
0xDC	Volkswagen ID.4
0xF0	Rivian

CAN message definitions vary by manufacturer, model, and year. The vehicles listed above may not work with every available model year.

All values not listed in the table above are reserved for future use.

3.1.6.4 Software Wheel Measurement Configuration (WheelConfig)

The `WheelConfig` payload is intended for use on vehicles where wheel speed or angle (tick) data is received via software, either using FusionEngine measurement messages, or from another software mechanism such as a vehicle CAN bus. Wheel data may be differential (measurements from each individual wheel), or scalar (a single speed measurement for the vehicle body).

When using software wheel data, you must also specify 3.1.6.3 Vehicle Details and Physical Dimensions (VehicleDetails).

See section 3.5 Sensor Measurement Input Messages for descriptions of the messages used to send wheel measurements to the device.

Note: Do not use this message for vehicles using a hardware wheel tick voltage signal. Instead, use 3.1.6.5 Hardware Wheel Tick Configuration (HardwareTickConfig).

Field	Data Type	Description

Wheel Sensor Type	`WheelSensorType (u8)`	The type of vehicle/wheel speed measurements produced by the vehicle.
Applied Speed Type	`AppliedSpeedType (u8)`	The type of vehicle/wheel speed measurements to be applied to the navigation solution.
Steering Type	`SteeringType (u8)`	Indication of which of the vehicle's wheels are steered.
Reserved	`u8`	*Reserved for future use.*
Wheel Update Interval	`f32`	The nominal rate at which wheel speed measurements will be provided in seconds. Note: this parameter is required when configured for software wheel speed measurements.
Wheel Tick Output Update Interval	`f32`	The nominal rate at which wheel tick measurements will be used by the navigation engine (in seconds). Note: For most system configurations, it is recommended to set this value to `NAN`, in which the device will select the appropriate setting.
Steering Ratio	`f32`	The ratio between the angle of the steering wheel and the angle of the wheels on the ground.
Meters/Tick Scale Factor	`f32`	The scale factor to convert wheel encoder ticks to distance in meters/tick. Used for `TICKS` and `VEHICLE_TICKS` wheel sensor types only.
Wheel Tick Max Value	`u32`	The maximum value before the wheel tick measurement will roll over. Used for `TICKS` and `VEHICLE_TICKS` wheel sensor types only. Rollover behavior depends on the value of the Wheel Ticks Signed setting. For example, a maximum value of 10 will work as follows: • Wheel Ticks Signed (true): [-11, 10] • Wheel Ticks Signed (false): [0, 10] Signed values are assumed to be asymmetric, consistent with a typical 2's complement rollover.

Wheel Ticks Signed	u8	1 if the wheel reported wheel tick measurements should be interpreted as signed integers, or 0 if they should be interpreted as unsigned integers. Used for TICKS and VEHICLE_TICKS wheel sensor types only.
Wheel Ticks Always Increase	u8	1 if the wheel tick measurements increase by a positive amount when driving forward or backward. 0 if wheel tick measurements decrease when driving backward. Used for TICKS and VEHICLE_TICKS wheel sensor types only.
Reserved	u8[2]	*Reserved for future use.*

WheelSensorType (u8):

Name	Value	Description
NONE	0x0	Wheel/vehicle speed data not available.
Reserved	0x1	*Reserved for future use.*
TICKS	0x2	Individual rotation angle measurements for multiple wheels, reported as accumulated encoder ticks. Will be scaled to meters using the Tick/Meter Scale Factor.
WHEEL_SPEED	0x3	Individual speed measurements for multiple wheels, reported in meters/second.
VEHICLE_SPEED	0x4	A single value indicating vehicle speed in meters/second.
VEHICLE_TICKS	0x5	A single wheel rotation angle, reported as accumulated encoder ticks.
Reserved	0x6-0xFF	*Reserved for future use.*

AppliedSpeedType (u8):

Name	Value	Description
NONE	0x0	Speed data not applied to the navigation solution.
REAR_WHEELS	0x1	Apply rear wheel speed data. Recommended for most use cases.
FRONT_WHEELS	0x2	Apply front wheel speed data.
FRONT_AND_REAR_WHEELS	0x3	Apply both front and rear wheel speed data.
VEHICLE_BODY	0x4	Apply aggregate vehicle speed data.

Reserved	0x5-0xFF	*Reserved for future use.*

SteeringType (u8):

Name	Value	Description
UNKNOWN	0x0	Steered wheels unknown/not used.
FRONT	0x1	Front wheels are steered.
FRONT_AND_REAR	0x2	Front and rear wheels are steered.
Reserved	0x3-0xFF	*Reserved for future use.*

3.1.6.5 Hardware Wheel Tick Configuration (HardwareTickConfig)

The `HardwareTickConfig` payload is intended for use on vehicles with a physical voltage signal, generated by a wheel encoder, producing a series of voltage pulses (encoder ticks) as the vehicle's wheel rotates. These ticks will be captured by the device on an input pin and used to indicate vehicle speed.

Additionally, an optional voltage signal may be provided to indicate vehicle direction. **If this signal is not connected to a real pulse (even if pulled up/down), the `TickDirection` field MUST be set to OFF otherwise there will be substantial errors in dead reckoning.**

Note: DO NOT use this configuration for vehicles using software wheel speed/tick information, including data sent using FusionEngine messages or a vehicle CAN bus. Instead, use 3.1.6.4 Software Wheel Measurement Configuration (WheelConfig).

Field	Data Type	Description
Tick Mode	`TickMode (u8)`	The signal edge to use when capturing the incoming wheel tick voltage signal. If enabled, the device will accumulate ticks received on the I/O pin and use them as an indication of vehicle speed. If specified, you must also specify the Meters/Tick Scale Factor to indicate the mapping of wheel tick encoder angle to tire circumference. All other wheel tick-related parameters such as tick capture rate, rollover value, etc. will be set internally.
Tick Direction	`TickDirection (u8)`	The way to interpret the optional direction voltage signal. If set to `OFF`, the incoming ticks will be treated as unsigned, meaning the tick count will continue to increase in either direction of travel. If not set to `OFF`, a second direction I/O pin will be used to indicate the direction of travel and the accumulated tick count will increase/decrease accordingly.
Reserved	`u8`	*Reserved for future use.*
Meters/Tick Scale Factor	`f32`	The scale factor to convert wheel encoder ticks to distance in meters/tick. Used for `TICKS` and `VEHICLE_TICKS` wheel sensor types only.

`TickMode (u8)`:

Name	Value	Description
`OFF`	`0x0`	Hardware wheel tick capture disabled.
`RISING_EDGE`	`0x1`	Capture a tick on the rising edge of the incoming pulse.
`FALLING_EDGE`	`0x2`	Capture a tick on the falling edge of the incoming pulse.
Reserved	`0x3-0xFF`	*Reserved for future use.*

`TickDirection` (u8):

Name	Value	Description
`OFF`	`0x0`	Vehicle direction signal disabled.
`FORWARD_ACTIVE_HIGH`	`0x1`	Assume vehicle is moving forward when direction signal voltage is high, and backward when voltage is low.
`FORWARD_ACTIVE_LOW`	`0x2`	Assume vehicle is moving forward when direction signal voltage is low, and backward when voltage is high.
Reserved	`0x3-0xFF`	*Reserved for future use.*

3.1.6.6 Interface Config Submessage (InterfaceConfigSubmessage)

Message used to configure parameters for a particular I/O interface on the device such as baud rate, TCP port, enabling diagnostic output, etc.

Field	Data Type	Description
Interface ID	`InterfaceID (u32)`	The ID of the interface to be configured or queried. Note that type *ALL* is not supported.
Interface Config Type	`InterfaceConfigType (u8)`	The setting to be configured.
Reserved	`u8[3]`	*Reserved for future use.*
Interface Config Payload Data	*Variable*	The payload definition for each `InterfaceConfigType` is listed in the table below.

`InterfaceConfigType (u8)`:

Specifies the configuration parameter to be set/queried for a given interface.

Name	Value	Payload Format	Description
`OUTPUT_DIAGNOSTICS_MESSAGES`	1	`bool`	Enable/disable output of diagnostic data on this interface. Note this will override the message rate/off settings for some FusionEngine messages.
`BAUD_RATE`	2	`u32`	Serial Baud Rate in bits/second. Valid for serial transports.
`REMOTE_ADDRESS`	3	`char[64]`	Configure the network address for a client to connect to, or the path to a local serial device or UNIX domain socket file. Valid for TCP, UDP, UNIX, and serial transports. String **must** be null-terminated.
`PORT`	4	`u16`	The network port for a client or server. Valid for TCP, UDP, and WebSocket transports.
`ENABLED`	5	`bool`	Enable/disable this interface.

DIRECTION	6	TransportDirection (u8)	Set the interface direction (client or server). Valid for TCP, UNIX, and WebSocket transports.
UNIX_SOCKET_TYPE	7	UNIXSocketType (u8)	Set the UNIX domain socket type (streaming, datagram, or sequenced). Valid for UNIX transports.

`TransportDirection (u8):`
Describes the direction (client/server) of an interface, where applicable.

Name	Value	Description
Invalid	0x0	Not used
Server	0x1	A server that listens for one or more incoming connections.
Client	0x2	A client that connects to a specified remote server.

`UNIXSocketType (u8):`
Specifies how data is transmitted when using a UNIX domain socket.

Name	Value	Description
Invalid	0x0	Not used
Stream	0x1	Operate in connection-oriented streaming mode and do not preserve message boundaries (similar to TCP).
Datagram	0x2	Operate in datagram mode, preserving message boundaries but not maintaining client connections (similar to UDP).
Sequenced Packet	0x3	Operate in sequenced packet mode, which is both connection-oriented and preserves message boundaries.

3.1.6.7 Ionosphere Delay Model Configuration (`IonosphereConfig`)

Message used to configure the ionospheric delay model.

Field	Data Type	Description
Ionosphere Delay Model	`IonoDelayModel (u8)`	The model used to predict ionospheric delays.
Reserved	`u8[3]`	*Reserved for future use.*

`IonoDelayModel` (u8):

Name	Value	Description
`AUTO`	`0x0`	Uses the best available ionospheric model.
`OFF`	`0x1`	Explicitly disables an ionospheric model. This is not recommended except for use in simulation.
`KLOBUCHAR`	`0x2`	Uses the Klobuchar ionospheric model.
`SBAS`	`0x3`	Uses the SBAS gridded ionospheric model (when available).
Reserved	`0x4-0xFF`	*Reserved for future use.*

3.1.6.8 Troposphere Delay Model Configuration (`TroposphereConfig`)

Message used to configure the tropospheric delay model.

Field	Data Type	Description
Troposphere Delay Model	`TropoDelayModel (u8)`	The model used to predict tropospheric delays.
Reserved	`u8[3]`	*Reserved for future use.*

`TropoDelayModel` (u8):

Name	Value	Description
`AUTO`	`0x0`	Uses the best available tropospheric model.
`OFF`	`0x1`	Explicitly disables a tropospheric model. This is not recommended except for use in simulation.
`SAASTAMOINEN`	`0x2`	Uses the Saastamoinen ionospheric model.
Reserved	`0x3-0xFF`	*Reserved for future use.*

3.1.6.9 LBandConfig

Configuration of L-band demodulator parameters.

Field	Data Type	Description
Center Frequency	`f64`	The center frequency of the L-band (Hz).
Search Window	`f32`	The size of the signal acquisition search space (in Hz) around the center frequency. For example, a value of 6000 will search +/- 3 kHz around the center frequency
Filter Data By Service ID	`bool`	Determines whether to only output data frames with the configured `Service ID`. If not, output all decoded frames.
Use Descrambler	`bool`	Enable/Disable the descrambler.
PMP Service ID	`u16`	The PMP service ID of the provider.
PMP Unique Word	`u64`	The PMP unique word used by provider.
Data Rate	`u16`	The transmitted data rate (bps).
Descrambler Initialization Value	`u16`	The initialization value for the PMP descrambling vector.

3.1.7 GetConfigMessage (13101)

Message ID: 13101

Protocol Version: 2
Message Version: 0

Description: Query the value of a user configuration parameter.

Field	Data Type	Description
Parameter Type	`ConfigType (u16)`	The parameter to be queried (see 3.1.6 SetConfig (13100)).
Configuration Source	`ConfigurationSource (u8)`	The data source to be queried (active, saved, etc.).
Reserved	`u8`	*Reserved for future use.*

`ConfigurationSource (u8):`

Configuration Source	Value	Description
`ACTIVE`	`0x0`	Active configuration settings currently in use by the device.
`SAVED`	`0x1`	Settings currently saved in persistent storage.

Expected Response(s):

3.2.2 ConfigResponseMessage (13103)

3.1.8 SaveConfigMessage (13102)

Message ID: 13102

Protocol Version: 2
Message Version: 0

Description: Save or reload configuration settings.

Field	Data Type	Description
Save Action	`SaveAction (u8)`	Indicates the action to perform on the saved configuration.
Reserved	`u8[3]`	*Reserved for future use.*

`SaveAction (u8)`:

Save Action	Value	Description
Save	`0x0`	Save all active parameters to persistent storage.
Revert to Saved	`0x1`	Revert the active configuration to previously saved values.
Revert to Default	`0x2`	Reset both the active and saved configuration to default values.

Expected Response(s):

3.2.1 CommandResponse (13000)

Example Usage

Example 1: Save configuration changes to persistent storage

```
2e31 0000 c48c 1ee8 0200 2e33 0000 0000
0400 0000 0000 0000 0000 0000
```

Example 2: Reset configuration settings to factory defaults

```
2e31 0000 4f44 1742 0200 2e33 0000 0000
0400 0000 0000 0000 0200 0000
```

This has the same effect as issuing a 3.1.2 ResetRequest (13002) with the bit mask set to `RESET_CONFIG`.

3.1.9 ImportDataMessage (13110)

Message ID: 13110

Protocol Version: 2
Message Version: 0

Description: Import data from the host to the device.

Field	Data Type	Description
Data Type	`DataType (u8)`	The type of data being imported.
Configuration Source	`Configuration Source (u8)`	The source of the parameter value (active, saved, etc.). If the Data Type parameter does not separate active and saved data, this will be ignored.
Reserved	`u8[2]`	*Reserved for future use.*
Data Version	`DataVersion (u16)`	Version of the data contents.
Reserved	`u8[4]`	*Reserved for future use.*
Data Length	`u32`	The number of bytes to update.
Value	*Variable*	Array of bytes for the data contents.

3.1.10 ExportDataMessage (13111)

Message ID: 13222

Protocol Version: 2
Message Version: 0

Description: Export data from the device to the host.

Field	Data Type	Description
Data Type	`DataType (u8)`	The type of data to be exported.
Configuration Source	`Configuration Source (u8)`	The source of the parameter value (active, saved, etc.). If the Data Type parameter does not separate active and saved data, this will be ignored.
Reserved	`u8[2]`	*Reserved for future use.*

3.1.11 SetMessageRate (13220)

Message ID: 13220

Protocol Version: 2
Message Version: 0

Description: Enable/disable an individual message on a specified output interface or configure its output rate.

Some messages can only be generated on change, and do not support rate control requests.

Note: If the `DIAGNOSTICS_ENABLED` configuration setting is set for the interface being configured, the specified rate may be temporarily overridden. This can be verified using the GetMessageRate (13221) command.

For a list of `u16` message IDs corresponding with supported NMEA-0183 messages, see 2.8 NMEA-0183 Message Types.

Field	Data Type	Description
Interface ID	`InterfaceID (u32)`	The ID of the interface to be configured.
Protocol Type	`ProtocolType (u8)`	The message protocol to be configured. If `ALL` (255) is specified, set the rate for all protocols and messages on the specified interface.
Flags	`u8`	A bitmask of flags to modify the command (see the table below).
Message ID	`u16`	The ID of the message type to be configured. If `ALL` (65535) is specified, set the rate for all supported messages for the specified protocol.
Message Rate	`MessageRate (u8)`	The desired output rate or on/off status. If the rate is an interval, and Message ID or Protocol Type is set to a wildcard, the command will only affect messages that are currently enabled. To modify all messages, including disabled messages, set the `FLAG_INCLUDE_DISABLED_MESSAGES` flag.
Reserved	`u8[3]`	*Reserved for future use.*

`ProtocolType (u8)`:
The framing protocol of a message.

Protocol	Value
Invalid	`0x0`
FusionEngine	`0x1`

NMEA	0x2
RTCM	0x3
ALL	0xFF

Flag bit definitions:

Flag	Bit	Value	Description
APPLY_AND_SAVE	0	0x01	Setting this bit saves the updated configuration to persistent storage immediately on success.
INCLUDE_DISABLED_MESSAGES	1	0x02	Setting this bit applies bulk interval changes to all supported messages instead of only currently enabled messages.
Reserved	2-7		Reserved for future use.

`MessageRate` (u8):

Message Rate	Value
Off	0x0
On Change	0x1
10 ms	0x2
20 ms	0x3
40 ms	0x4
50 ms	0x5
100 ms	0x6
200 ms	0x7
500 ms	0x8
1 second	0x9
2 seconds	0xA
5 seconds	0xB
10 seconds	0xC
30 seconds	0xD
60 seconds	0xE
Device Default	0xFF

Expected Response(s):

3.2.1 CommandResponse (13000)

Example Usage

Example 1: Enable FusionEngine PoseMessage output at its max rate on serial port 1 (UART1)

```
2e31 0000 6a84 7724 0200 a433 0000 0000
0c00 0000 0000 0000 0101 0000 0100 1027
0100 0000
```

Example 2: Enable FusionEngine PoseMessage output at its max rate on serial port 1 (UART1) then save the change to persistent memory.

```
2e31 0000 de8f 0082 0200 a433 0000 0000
0c00 0000 0000 0000 0101 0000 0101 1027
0100 0000
```

Example 3: Set all message rates on all interfaces to their factory default values.

```
2e31 0000 aa8d 150d 0200 a433 0000 0000
0c00 0000 0000 0000 ff00 0000 ff02 ffff
ff00 0000
```

Note that if FLAG_INCLUDE_DISABLED_MESSAGES is not set, only the messages already enabled will be modified.

Example 4: On UART 1, set all enabled NMEA messages that support rate control to output at 1Hz.

```
2e31 0000 a936 db31 0200 a433 0000 0000
0c00 0000 0000 0000 0101 0000 0200 ffff
0900 0000
```

3.1.12 GetMessageRate (13221)

Message ID: 13221

Protocol Version: 2
Message Version: 0

Description: Query the configuration for the requested message type on the specified interface.

Field	Data Type	Description
Interface ID	`InterfaceID (u32)`	The ID of the interface to be queried.
Protocol Type	`ProtocolType (u8)`	The message protocol to be queried. If the `ALL` (255) is specified, return the rate for all protocols and messages on the specified interface.
Configuration Source	`ConfigurationSource (u8)`	The data source to be queried.
Message ID	`u16`	The ID of the message type to be queried. If `ALL` (65535) is specified, return the rate for all supported messages for the specified protocol.

`ConfigurationSource (u8)`:

Configuration Source	Value	Description
`ACTIVE`	`0x0`	Active configuration settings currently in use by the device.
`SAVED`	`0x1`	Settings currently saved in persistent storage.

Expected Response(s):

3.2.4 MessageRateResponse (13222)

3.2 Command Response Messages (Output)

These are messages sent by the device in response to requests or commands from the host.

3.2.1 CommandResponseMessage (13000)

Message ID: 13000

Protocol Version: 2
Message Version: 0

Description: Response to user commands.

Field	Data Type	Description
Source Sequence Number	`u32`	The sequence number of the command that triggered this response.
Response Code	`Response (u8)`	Indicates the type of the response.
Reserved	`u8[3]`	*Reserved for future use.*

`Response (u8)`:

Response Type	Response Code
OK	0
Unsupported Message Version	1
Unsupported Feature	2
Value Error	3
Insufficient Space	4
Execution Failure	5
Inconsistent Payload Length	6
Data Corrupted	7
No Data Stored	8
Unavailable	9
Unsupported Interface	10

3.2.2 ConfigResponseMessage (13103)

Message ID: 13103

Protocol Version: 2
Message Version: 0

Description: The response to a 3.1.7 GetConfigMessage (13101).

If an invalid or unsupported parameter was requested, the Response Code will indicate that the parameter is unsupported/invalid.

Field	Data Type	Description
Configuration Source	`Configuration Source (u8)`	The source of the parameter value (active, saved, etc.).
Active Differs From Saved	`u8`	1 if the active value for this parameter differs from the value saved in persistent storage. 0 if the active and saved values are the same.
Parameter Type	`ConfigType (u16)`	The type of parameter being returned. Will always reflect the `ConfigType` in the corresponding `GetConfigMessage`.
Response Code	`Response (u8)`	Indicates the status of the response on success or failure (see 3.2.1 CommandResponse (13000)).
Reserved	`u8[3]`	*Reserved for future use.*
Value Length	`u32`	The length of the value to follow (in bytes).
Value	*Variable*	The current value of the requested parameter. The format of the value is listed in the `ConfigType` table in 3.1.6 SetConfigMessage (13100)

`ConfigurationSource (u8)`:

Configuration Source	Value	Description
`ACTIVE`	`0x0`	Active configuration settings currently in use by the device.
`SAVED`	`0x1`	Settings currently saved in persistent storage.

3.2.3 PlatformStorageDataMessage (13113)

Message ID: 13113

Protocol Version: 2
Message Version: 3

Description: Message for reporting platform storage data.

Field	Data Type	Description
Data Type	`DataType (u8)`	The type of data contained in this message.
Response	`Response (u8)`	The status of the specified data type on the device.
Configuration Source	`ConfigurationSource (u8)`	The source of the returned contents (active settings, saved, etc.). If the Data Type parameter does not distinguish between active and saved data, this will be ignored.
Flags	`u8`	A field describing additional details about the returned contents. The definition of this field depends on the Data Type.
Data Version	`DataVersion (u16)`	Version of the data contents.
Data Length	`u32`	The number of bytes in data contents
Value	*Variable*	Array of bytes for the data contents.

Flag definitions (DataType::USER_CONFIG):

Flag	Value	Description
Platform Not Specified	0	Platform type not indicated for the returned data.
POSIX Platform	1	User configuration data for FusionEngine on a POSIX platform.
Embedded Platform	2	User configuration data for FusionEngine on an embedded device.
Embedded SSR Client	3	User configuration data for Point One SSR Client on an embedded device.
SSR Client	4	User configuration data for Point One SSR Client desktop application or library.

3.2.4 MessageRateResponse (13222)

Message ID: 13222

Protocol Version: 2
Message Version: 0

Description: Response to a 3.1.12 GetMessageRate (13221) message.

Field	Data Type	Description
Configuration Source	`ConfigurationSource (u8)`	The source of the parameter value (active, saved, etc.).
Response Code	`Response (u8)`	Indicates the type of the response.
Number of Rates	`u16`	The number of rates being reported in this message.
Interface ID	`InterfaceID (u32)`	The ID of the interface described by this message.
Rates	`MessageRateResponseEntry [Number of Rates]`	A repeated field containing Number of Rates entries, one for each message in the response.

3.2.7.1 Message Rate Response Entry (`MessageRateResponseEntry`)

Field	Data Type	Description
Protocol Type	`ProtocolType (u8)`	The message protocol.
Flags	`u8`	A bitmask of flags to modify the command (see the table below).
Message ID	`u16`	The ID of the message type.
Configured Rate	`MessageRate (u8)`	The configured output rate or on/off status.
Effective Rate	`MessageRate (u8)`	The actual output rate for this message, factoring in effects of additional configuration settings that may override the configured rate such as enabling diagnostic output. In general, this will be equal to the configured rate.
Reserved	`u8[2]`	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
`ACTIVE_DIFFERS_FROM_SAVED`	`0`	0x01	Flag to indicate the active value for this configuration differs from the value saved to persistent memory.

Reserved	1-7		*Reserved for future use.*

3.2.5 SupportedIOInterfacesMessage (13223)

Message ID: 13223

Protocol Version: 2
Message Version: 0

Description: List the available interfaces supported by the device.

The maximum number of each type of interface supported by a given device depends on the type of device and the FusionEngine software version. For example, a Point One Atlas device might support up to 5 configurable TCP sockets (client or server), along with other transport types, while an embedded Quectel LG69T-AP device only supports 2 fixed serial UART connections.

This message is sent in response to a 3.1.1 MessageRequest (13001) message.

Field	Data Type	Description
Number of Interfaces	`u8`	Number of interfaces, N, reported by this message
Reserved	`u8[7]`	*Reserved for future use.*
Interface Descriptions	InterfaceID[N]	Descriptions of each interface supported by the device. E.g., Serial 1, Serial 2, TCP, etc.

3.3 Navigation Solution Messages (Output)

These messages contain the navigation solution calculated by FusionEngine, along with other related information (GNSS signal tracking and usage status, etc.).

3.3.1 PoseMessage (10000)

Message ID: 10000

Protocol Version: 2
Message Version: 2

Description: Position, velocity, and attitude (orientation) solution.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
GPS Time	`Timestamp (8B)`	The GPS time of the solution, if available.
Solution Type	`SolutionType (u8)`	The type of the position solution (see below).
Flags	`u8`	A bitmask of flags associated with the pose data. *Added in PoseMessage version 1.2.*
Undulation	`s16`	The geoid undulation at the current location in cm, defined as the height of the MSL geoid above the ellipsoid: Orthometric (MSL) Altitude = Ellipsoid Altitude - Undulation where the ellipsoid altitude is in the Altitude field below. Set to -32768 if invalid. *Added in PoseMessage version 1.1.*
Latitude	`f64`	The geodetic latitude in degrees, expressed using the WGS-84 reference ellipsoid.
Longitude	`f64`	The geodetic longitude in degrees, expressed using the WGS-84 reference ellipsoid.
Altitude	`f64`	The geodetic altitude (height above the ellipsoid) in meters, expressed using the WGS-84 reference ellipsoid.
Position Std Dev (East)	`f32`	The standard deviation of the position in the East direction in meters.
Position Std Dev (North)	`f32`	The standard deviation of the position in the North direction in meters.
Position Std Dev (Up)	`f32`	The standard deviation of the position in the Up direction in meters.

Yaw	`f64`	The platform yaw in degrees. Positive yaw indicates a left turn, negative yaw indicates a right turn.
Pitch	`f64`	The platform pitch in degrees. Positive pitch points the nose of the vehicle down, negative pitch points the nose up.
Roll	`f64`	The platform roll in degrees. Positive roll indicates a roll toward the right, negative roll is a roll to toward the left.
Yaw Std Dev	`f32`	Yaw standard deviation in degrees.
Pitch Std Dev	`f32`	Pitch standard deviation in degrees.
Roll Std Dev	`f32`	Roll standard deviation in degrees.
Forward Velocity	`f64`	Body forward velocity in meters/second. Set to `NAN` if body orientation information is not available.
Left Velocity	`f64`	Body left velocity in meters/second. Set to `NAN` if body orientation information is not available.
Up Velocity	`f64`	Body up velocity in meters/second. Set to `NAN` if body orientation information is not available.
Forward Velocity Std Dev	`f32`	Forward velocity standard deviation in meters/second.
Left Velocity Std Dev	`f32`	Left velocity standard deviation in meters/second.
Up Velocity Std Dev	`f32`	Up velocity standard deviation in meters/second.
Aggregate Protection Level	`f32`	The estimated 3D protection level in meters.
Horizontal Protection Level	`f32`	The estimated 2D horizontal protection level in meters.
Vertical Protection Level	`f32`	The estimated vertical level in meters.

`SolutionType (u8):`

Type	Value	Description
`Invalid`	`0x0`	Invalid, no position available.
`AutonomousGPS`	`0x1`	Standalone GNSS fix, no corrections data used.
`DGPS`	`0x2`	Differential GNSS pseudorange solution using a local RTK base station, SSR corrections data, or SBAS corrections data.
`Reserved`	`0x3`	*Reserved for future use.*
`RTK Fixed`	`0x4`	GNSS RTK solution with fixed integer carrier phase ambiguities (most accurate position solution).

RTK Float	0x5	GNSS RTK solution with floating point carrier phase ambiguities.
Integrate	0x6	Dead reckoned position, computed by integrating IMU data or an internal motion model.
Reserved	0x7-0x8	*Reserved for future use.*
Visual	0x9	Position solution using vision measurement inputs.
PPP	0xA	GNSS precise point positioning (PPP) pseudorange/carrier phase solution.
Reserved	0xB-xFF	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
STATIONARY	0	0x01	Set if the device is currently stationary.
Reserved	1-7		*Reserved for future use.*

3.3.2 GNSSInfoMessage (10001)

Message ID: 10001

Protocol Version: 2
Message Version: 1

Description: Information about the GNSS data used in the corresponding PoseMessage (10000).

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
GPS Time	`Timestamp (8B)`	The GPS time of the solution.
Leap Second	`u8`	The current UTC leap second (offset between UTC and GPS time), if known. Set to 0xFF if invalid.
Number of Satellites	`u8`	The number of satellites used in the current position solution.
Reserved	`u8[2]`	*Reserved*
Corrections Age	`u16`	The age of the most recently received GNSS corrections data (in 0.1 seconds). Set to 0xFFFF if invalid.
Baseline Distance	`u16`	The distance between the device and the GNSS corrections base station. Stored in units of 10 meters: `baseline_distance_m = baseline_distance * 10` Set to 0xFFFF if invalid.
Reference Station ID	`u32`	The ID of the differential base station, if used. Set to 0xFFFFFFFF if invalid.
GDOP	`f32`	The geometric dilution of precision (GDOP).
PDOP	`f32`	The position dilution of precision (PDOP).
HDOP	`f32`	The horizontal dilution of precision (HDOP).
VDOP	`f32`	The vertical dilution of precision (VDOP).
GPS Time Std Dev	`f32`	The standard deviation of the GPS time solution (i.e., the GPS clock bias estimate) in seconds.

3.3.3 GNSSSatelliteMessage (10002)

Message ID: 10002

Protocol Version: 2
Message Version: 1

Description: Information about the individual GNSS satellites used in the corresponding PoseMessage (10000) and GNSSInfoMessage (10001).

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
GPS Time	`Timestamp (8B)`	The GPS time of the solution.
Number Of Satellites	`u16`	The total number of available satellites (N).
Reserved	`u8[2]`	*Reserved for future use.*
Satellite Info Entries	`SatelliteInfo[N]`	A repeated instance of `SatelliteInfo` blocks describing the satellites used in the solution.

`SatelliteInfo:`

Field	Data Type	Description
Satellite Type	`SatelliteType (u8)`	The GNSS constellation to which this satellite belongs (see Section 2.6.5)
PRN	`u8`	The satellite identifier (PRN).
Usage Mask	`u8`	A bitmask indicating how this satellite was used in the position solution (see below).
C/N0	`u8`	The carrier-to-noise density ratio (C/N0) for the L1 signal on the satellite in dB-Hz. Set to 0 if invalid. If the satellite is not tracking L1 (or the near-L1 equivalent signal for the constellation), but is tracking a signal on another frequency, that signal's C/N0 value will be reported.
Azimuth	`f32`	The azimuth of the satellite in degrees.
Elevation	`f32`	The elevation of the satellite in degrees.

`Usage mask (u8):`

Bit	Description
0	One or more signals from this satellite used in the position solution.
1-7	*Reserved for future use.*

3.3.4 PoseAux (10003)

Message ID: 10003

Protocol Version: 2
Message Version: 0

Description: Auxiliary position solution information.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
Position Std Dev (Forward)	`f32`	The standard deviation of the position resolved in the vehicle body forward direction in meters.
Position Std Dev (Left)	`f32`	The standard deviation of the position resolved in the vehicle body left direction in meters.
Position Std Dev (Up)	`f32`	The standard deviation of the position resolved in the vehicle body up direction in meters.
Position Covariance	`f64[9]`	The 3x3 position covariance matrix (in meters^2), resolved in the local ENU frame and stored in row-major order.
Attitude Quaternion	`f64[4]`	The platform body orientation with respect to the local ENU frame, represented as a quaternion with the scale component last (x, y, z, w).
East Velocity	`f64`	Platform velocity in the East direction in meters/second.
North Velocity	`f64`	Platform velocity in the North direction in meters/second.
Up Velocity	`f64`	Platform velocity in the Up direction in meters/second.
East Velocity Std Dev	`f32`	East velocity standard deviation in meters/second.
North Velocity Std Dev	`f32`	North velocity standard deviation in meters/second.
Up Velocity Std Dev	`f32`	Up velocity standard deviation in meters/second.

3.3.5 CalibrationStatus (10004)

Message ID: 10004

Protocol Version: 2
Message Version: 1S

Description: Current device calibration status.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
Calibration Stage	`Calibration Stage (u8)`	The current calibration stage (see below).
Reserved	`u8[3]`	*Reserved for future use.*
Yaw Mounting Angle	`f32`	The estimated IMU yaw mounting angle offset in degrees.
Pitch Mounting Angle	`f32`	The estimated IMU pitch mounting angle offset in degrees.
Roll Mounting Angle	`f32`	The estimated IMU roll mounting angle offset in degrees.
Yaw Std Dev	`f32`	The estimated yaw mounting angle standard deviation in degrees.
Pitch Std Dev	`f32`	The estimated pitch mounting angle standard deviation in degrees.
Roll Std Dev	`f32`	The estimated roll mounting angle standard deviation in degrees.
Travel Distance	`f32`	The accumulated calibration travel distance in meters.
Reserved	`u8[24]`	*Reserved for future use.*
State Verified	`u8`	1 if the navigation engine state has been verified after initialization, 0 otherwise.
Reserved	`u8[3]`	*Reserved for future use.*
Gyro Bias Percent	`u8`	The completion percentage for IMU gyro bias estimation, stored in units of 0.5% (range [0, 200]).
Accel Bias Percent	`u8`	The completion percentage for IMU accelerometer bias estimation, stored in units of 0.5% (range [0, 200]).
Mounting Angle Bias Percent	`u8`	The completion percentage for IMU mounting angle estimation, stored in units of 0.5% (range [0, 200]).
Reserved	`u8[5]`	*Reserved for future use.*

Min Travel Distance	`f32`	The minimum travel distance needed to complete mounting angle calibration in meters.
Max Yaw Std Dev	`f32`	The maximum yaw mounting angle standard deviation (in degrees), above which calibration is incomplete.
Max Pitch Std Dev	`f32`	The maximum pitch mounting angle standard deviation (in degrees), above which calibration is incomplete.
Max Roll Std Dev	`f32`	The maximum roll mounting angle standard deviation (in degrees), above which calibration is incomplete.

`CalibrationStage (u8):`

Stage	Value	Description
Unknown	`0x0`	Calibration stage not known.
Mounting Angle	`0x1`	Estimating IMU mounting angles.
Reserved	`0x2-0xFE`	*Reserved for future use.*
Done	`0xFF`	Calibration complete.

3.3.6 RelativeENUPosition (10005)

Message ID: 10005

Protocol Version: 2
Message Version: 0

Description: Rover position relative to a local RTK base station, expressed in the ENU frame centered at the base station.

Note: this message represents the relationship between the navigation engine's position solution and the location of a nearby RTK base station. It is not used to convey vehicle orientation measurements when using multiple GNSS antennas. See 3.6.2 RawGNSSAttitudeOutput (11006) instead.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
GPS Time	`Timestamp (8B)`	The GPS time of the solution.
Solution Type	`SolutionType (u8)`	The type of the position solution (see 3.3.1 Pose (10000)).
Reserved	`u8[3]`	*Reserved for future use.*
Reference Station ID	`u32`	The ID of the differential base station, if used. Set to 0xFFFFFFFF if invalid.
East Position	`f64`	The position offset in the East direction relative to the local base station position (in meters).
North Position	`f64`	The position offset in the North direction relative to the local base station position (in meters).
Up Position	`f64`	The position offset in the Up direction relative to the local base station position (in meters).
Position Std Dev (East)	`f32`	The standard deviation of the position in the East direction in meters.
Position Std Dev (North)	`f32`	The standard deviation of the position in the North direction in meters.
Position Std Dev (Up)	`f32`	The standard deviation of the position in the Up direction in meters.

3.4 Corrected Sensor Measurement Output Messages

These are the primary sensor outputs that are compensated, calibrated, and filtered outputs of the system. In general these are the best outputs to use for upstream processing. For raw (uncorrected) measurements, see Section 3.6.

3.4.1 IMUOutput (11000)

Message ID: 11000

Protocol Version: 2
Message Version: 0

Description: Corrected IMU measurement data.

The data contained in this message has been corrected for estimated accelerometer and gyro errors, including biases and scale factors, and has been rotated into the vehicle body frame from the original IMU orientation.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
X Acceleration	`f64`	The vehicle X acceleration in meters/second^2.
Y Acceleration	`f64`	The vehicle Y acceleration in meters/second^2.
Z Acceleration	`f64`	The vehicle Z acceleration in meters/second^2.
X Acceleration Std Dev	`f64`	The standard deviation of X acceleration in meters/second^2.
Y Acceleration Std Dev	`f64`	The standard deviation of Y acceleration in meters/second^2.
Z Acceleration Std Dev	`f64`	The standard deviation of Z acceleration in meters/second^2.
X Rotation Rate	`f64`	The vehicle X axis gyroscope rate of rotation in radians/second.
Y Rotation Rate	`f64`	The vehicle Y axis gyroscope rate of rotation in radians/second.
Z Rotation Rate	`f64`	The vehicle Z axis gyroscope rate of rotation in radians/second.
X Rotation Std Dev	`f64`	The standard deviation of the X rate of rotation in radians/second.
Y Rotation Std Dev	`f64`	The standard deviation of the Y rate of rotation in radians/second.
Z Rotation Std Dev	`f64`	The standard deviation of the Z rate of rotation in radians/second.

3.4.2 GNSSAttitudeOutput (11005)

Message ID: 11005

Protocol Version: 2
Message Version: 0

Description: Vehicle orientation measured using the relative position relationship of two or more GNSS antennas.

When lever arms are configured correctly for each antenna, the values in this message will be corrected to account for the angular offset between the two antennas such that the reported yaw, pitch, and roll values represent the vehicle body. If the antennas are not installed inline with the vehicle +X axis and lever arms are not configured, the reported yaw, pitch, and roll values will not reflect the orientation of the vehicle. See GNSS Lever Arm and GNSS Aux Lever Arm in 3.1.6 SetConfigMessage (13100).

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Solution Type	`SolutionType (u8)`	`Invalid`: Heading not available `RTKFixed`: Heading available See section 3.3.1 PoseMessage (10000).
Reserved	`u8[3]`	*Reserved for future use.*
Flags	`u8`	A bitmask of flags associated with the solution.
Yaw	`f32`	The measured yaw angle (in degrees).
Pitch	`f32`	The measured pitch angle (in degrees).
Roll	`f32`	The measured roll angle (in degrees).
Yaw Std Dev	`f32`	Yaw standard deviation (in degrees).
Pitch Std Dev	`f32`	Pitch standard deviation (in degrees).
Roll Std Dev	`f32`	Roll standard deviation (in degrees).
Baseline Distance	`f32`	The distance between the primary and secondary antennas (in m). Set to NaN if unavailable.
Baseline Std Dev	`f32`	The standard deviation of the baseline distance (in m).

Flag bit definitions:

Flag	Bit	Value	Description
`Reserved`	`0-7`		*Reserved for future use.*

3.4.3 WheelSpeedOutput (11135)

Message ID: 11135

Protocol Version: 2
Message Version: 0

Description: This message is an output from the device that contains the speed of each individual wheel, after applying any estimated corrections (wheel scale factor, sign, etc).

Wheel odometry may be received via a software input from a host machine, a vehicle CAN bus, or a hardware voltage signal (wheel ticks). The Data Source field indicates which type of data source was provided to the system.

Note: When odometry is provided using hardware wheel ticks, the output rate of this message may differ from the wheel tick input rate. For higher accuracy applications, FusionEngine may integrate tick counts over longer time intervals to improve performance.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	P1 Time corresponding to the sensor measurement.
Sensor Data Source	`SensorDataSource (u8)`	The source of the incoming data, if known.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Flags	`u8`	Bitmask of flags associated with measurement data.
Reserved	`u8`	*Reserved for future use.*
Front Left Speed	`f32`	The front left wheel speed (in m/s). Set to NaN if unavailable.
Front Right Speed	`f32`	The front right wheel speed (in m/s). Set to NaN if unavailable.
Rear Left Speed	`f32`	The rear left wheel speed (in m/s). Set to NaN if unavailable.
Rear Right Speed	`f32`	The rear right wheel speed (in m/s). Set to NaN if unavailable.

Flag bit definitions:

Flag	Bit	Value	Description
`SIGNED`	0	0x01	Set to 1 if the speed values are signed (positive forward), 0 if unsigned.
Reserved	1-7		*Reserved for future use.*

3.4.4 VehicleSpeedOutput (11136)

Message ID: 11136

Protocol Version: 2
Message Version: 0

Description: This message is an output from the device that contains along-track speed of the vehicle (forward or backward) and applying any estimated corrections (scale factor, etc).

Odometry data may be received via software input from a host machine, a vehicle CAN bus, or a hardware voltage signal (encoder ticks). Data Source will indicate the data source providing the odometry measurements.

Note: When odometry is provided using hardware wheel ticks, the output rate of this message may differ from the wheel tick input rate. For higher accuracy applications, FusionEngine may integrate tick counts over longer time intervals to improve performance.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	P1 Time corresponding to the sensor measurement.
Sensor Data Source	`SensorDataSource (u8)`	The source of the incoming data, if known.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Flags	`u8`	Bitmask of flags associated with measurement data.
Reserved	`u8`	*Reserved for future use.*
Vehicle Speed	`f32`	Vehicle body velocity (in m/s). Set to NaN if unavailable.

Flag bit definitions:

Flag	Bit	Value	Description
`SIGNED`	0	0x01	Set to 1 if the speed value is signed (positive forward), 0 if unsigned.
Reserved	1-7		*Reserved for future use.*

3.5 Sensor Measurement Input Messages

For various sensor types, it is possible to send externally captured measurements to the device through software using defined FusionEngine measurement input messages. For example, a vehicle may measure wheel speed on an external processor and send the measurements to the FusionEngine device using a WheelSpeedInput message. Note that this is separate from the FusionEngine device capturing wheel tick voltage pulses in hardware or decoding incoming CAN data on its own.

In order to use software sensor measurements, you must set the corresponding configuration in the 3.1.6 SetConfigMessage (13100).

Software Wheel Speed Support

FusionEngine supports integration of several types of vehicle speed information to improve dead reckoning and navigation performance.

Wheel **speeds** are defined as rate sensors that have an indication of the velocity of each wheel. Wheel **ticks** are defined as counters that increment a fixed amount per rotation of the wheel, commonly found on anti-lock braking systems.

Vehicle messages are used to convey a single speed or tick value, indicating the forward velocity of the vehicle. **Wheel** messages are used to convey measurements for each individual wheel.

For best performance, we recommend differential rear wheel speed or tick data (e.g., left and right rear wheels) and vehicle gear position.

Only one type of speed or tick measurement should be provided to the device at a time.

3.5.1 IMUInput (11004)

Message ID: 11004

Protocol Version: 2
Message Version: 0

Description: Used to convey raw IMU measurements. See 3.1.5 SetConfigMessage (13100).

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Reserved	`u8[6]`	*Reserved for future use.*
Temperature	`i16`	The IMU temperature (in deg Celsius * 2e-7). Set to 0x7FFF if invalid.
X Acceleration	`i32`	The vehicle X acceleration in m/s^2 * 2e-16. Set to 0x7FFFFFFF if invalid.
Y Acceleration	`i32`	The vehicle Y acceleration in m/s^2 * 2e-16. Set to 0x7FFFFFFF if invalid.
Z Acceleration	`i32`	The vehicle Z acceleration in m/s^2 * 2e-16. Set to 0x7FFFFFFF if invalid.
X Rotation Rate	`i32`	The vehicle X rotation rate in radians/s * 2e-20. Set to 0x7FFFFFFF if invalid.
Y Rotation Rate	`i32`	The vehicleY rotation rate in radians/s * 2e-20. Set to 0x7FFFFFFF if invalid.
Z Rotation Rate	`i32`	The vehicle Z rotation rate in radians/s * 2e-20. Set to 0x7FFFFFFF if invalid.

3.5.2 WheelTickInput (11103)

Message ID: 11103

Protocol Version: 2
Message Version: 0

Description: Conveys the wheel encoder tick counts for one or more wheels. The number and type of wheels expected, and the interpretation of the tick count values varies by vehicle. You must configure vehicle dimensions and a description of the sensor configuration (number of wheels, etc.). See 3.1.5 SetConfigMessage (13100).

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Front Left Tick Count	`u32`	The front left wheel tick count.
Front Right Tick Count	`u32`	The front right wheel tick count.
Rear Left Tick Count	`u32`	The rear left wheel tick count.
Rear Right Tick Count	`u32`	The rear right wheel tick count.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Reserved	`u8[3]`	*Reserved for future use.*

3.5.3 VehicleTickInput(11104)

Message ID: 11104

Protocol Version: 2
Message Version: 0

Description: A single wheel encoder tick count that corresponds to forward vehicle movement. You must configure a description of the sensor configuration (tick rollover value, etc.). See SetConfigMessage (13100).

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Tick Count	`u32`	The current encoder tick count.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Reserved	`u8[3]`	*Reserved for future use.*

3.5.4 WheelSpeedInput (11105)

Message ID: 11105

Protocol Version: 2
Message Version: 0

Description: Used to convey the speed of each individual wheel on the vehicle. The number and type of wheels expected varies by vehicle. You must configure a description of the sensor configuration in SetConfigMessage (13100).

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Front Left Speed	`i32`	The front left wheel speed (in m/s * 2^-10). Set to `0x7FFFFFFF` if unavailable.
Front Right Speed	`i32`	The front right wheel speed (in m/s * 2^-10). Set to `0x7FFFFFFF` if unavailable.
Rear Left Speed	`i32`	The rear left wheel speed (in m/s * 2^-10). Set to `0x7FFFFFFF` if unavailable.
Rear Right Speed	`i32`	The rear right wheel speed (in m/s * 2^-10). Set to `0x7FFFFFFF` if unavailable.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Flags	`u8`	A bitmask of flags associated with the measurement data.
Reserved	`u8[2]`	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
`SIGNED`	0	0x01	Set to 1 if the speed values are signed (positive forward), 0 if unsigned.
Reserved	1-7		*Reserved for future use.*

3.5.5 VehicleSpeedInput (11106)

Message ID: 11106

Protocol Version: 2
Message Version: 0

Description: A single number (usually an average of the non-steered wheels) that corresponds to forward vehicle velocity. Note that only using a single wheel as “vehicle speed” can result in suboptimal performance while the vehicle is turning.

You must configure a description of the sensor configuration (tick rollover value, etc.). See SetConfigMessage (13100).

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Vehicle Speed	`i32`	The vehicle speed estimate (in m/s * 2^-10). Set to `0x7FFFFFFF` if unavailable.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Flags	`u8`	Bitmask of flags associated with measurement data.
Reserved	`u8[2]`	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
`SIGNED`	0	0x01	Set to 1 if the speed value is signed (positive forward), 0 if unsigned.
Reserved	1-7		*Reserved for future use.*

3.6 Raw (Uncorrected) Sensor Measurement Output Messages

Raw outputs are available for offboard processing. These messages are *not* compensated for any offset or calibration. The only additional processing is the calculation of the P1 Time, which is populated in the message's MeasurementDetails structure.

3.6.1 RawIMUOutput (11002)

Message ID: 11002

Protocol Version: 2
Message Version: 0

Description: Raw IMU measurement data.

This message is an output from the device containing raw IMU acceleration and rotation rate measurements. These measurements come directly from the sensor and do not have any corrections, calibration, or rotations applied.

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Reserved	`u8[6]`	*Reserved for future use.*
Temperature	`i16`	The IMU temperature (in Celsius * 2^-7). Set to 0x7FFF if invalid
X Acceleration	`i32`	Measured x acceleration (in meters/second^2 * 2^-16), resolved in the sensor measurement frame. Set to 0x7FFFFFFF if invalid
Y Acceleration	`i32`	Measured y acceleration (in meters/second^2 * 2^-16), resolved in the sensor measurement frame. Set to 0x7FFFFFFF if invalid
Z Acceleration	`i32`	Measured z acceleration (in meters/second^2 * 2^-16), resolved in the sensor measurement frame. Set to 0x7FFFFFFF if invalid
X Rotation Rate	`i32`	Measured x rate of rotation (in radians/second * 2^-20), resolved in the sensor measurement frame. Set to 0x7FFFFFFF if invalid.
Y Rotation Rate	`i32`	Measured y rate of rotation (in radians/second * 2^-20), resolved in the sensor measurement frame. Set to 0x7FFFFFFF if invalid.
Z Rotation Rate	`i32`	Measured z rate of rotation (in radians/second * 2^-20), resolved in the sensor measurement frame. Set to 0x7FFFFFFF if invalid.

3.6.2 RawGNSSAttitudeOutput (11006)

Message ID: 11006

Protocol Version: 2
Message Version: 0

Description: The relative position vector between two GNSS antennas in a multi-antenna device, used to measure the orientation of a vehicle. The coordinates are expressed in the local ENU reference frame.

See also 3.4.2 GNSSAttitudeOutput (11005).

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Solution Type	`SolutionType (u8)`	`Invalid`: Heading not available `RTKFixed`: Heading available See section 3.3.1 PoseMessage (10000).
Reserved	`u8[3]`	*Reserved for future use.*
Flags	`u32`	A bitmask of flags associated with the solution.
Relative Position (East)	`f32`	The East component of the vector from the primary antenna to the secondary antenna (in meters).
Relative Position (North)	`f32`	The North component of the vector from the primary antenna to the secondary antenna (in meters).
Relative Position (Up)	`f32`	The Up component of the vector from the primary antenna to the secondary antenna (in meters).
Position Std Dev (East)	`f32`	The standard deviation of the position in the East direction in meters.
Position Std Dev (North)	`f32`	The standard deviation of the position in the North direction in meters.
Position Std Dev (Up)	`f32`	The standard deviation of the position in the Up direction in meters.

Flag bit definitions:

Flag	Bit	Value	Description
Reserved	0-7		*Reserved for future use.*

3.6.3 RawWheelTickOutput (11123)

Message ID: 11123

Protocol Version: 2
Message Version: 0

Description: This message is an output from the device that contains wheel encoder tick counts for each individual wheel on the vehicle. These measurements come directly from the sensor, and do not have any corrections or calibration applied.

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Front Left Tick Count	`u32`	The front left wheel tick count.
Front Right Tick Count	`u32`	The front right wheel tick count.
Rear Left Tick Count	`u32`	The rear left wheel tick count.
Rear Right Tick Count	`u32`	The rear right wheel tick count.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Reserved	`u8[3]`	*Reserved for future use.*

3.6.4 RawVehicleTickOutput(11124)

Message ID: 11124

Protocol Version: 2
Message Version: 0

Description: This message is an output from the device that contains a wheel encoder tick count representing the along-track speed of the vehicle (forward/backward). This value comes directly from the sensor, and does not have any corrections or calibration applied.

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Tick Count	`u32`	The current encoder tick count.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Reserved	`u8[3]`	*Reserved for future use.*

3.6.5 RawWheelSpeedOutput (11125)

Message ID: 11125

Protocol Version: 2
Message Version: 0

Description: This message is an output from the device that contains the speed of each individual wheel on the vehicle. These measurements come directly from the sensor, and do not have any corrections or calibration applied.

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Front Left Speed	`i32`	Front left wheel speed (in `m/s * 2e-10`). Set to `0x7fffffff` if unavailable.
Front Right Speed	`i32`	Front right wheel speed (in `m/s * 2e-10`). Set to `0x7fffffff` if unavailable.
Rear Left Speed	`i32`	Rear left wheel speed (in `m/s * 2e-10`). Set to `0x7fffffff` if unavailable.
Rear Right Speed	`i32`	Rear right wheel speed (in `m/s * 2e-10`). Set to `0x7fffffff` if unavailable.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Flags	`u8`	Bitmask of flags associated with measurement data.
Reserved	`u8[2]`	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
`SIGNED`	0	0x01	Set to 1 if the speed values are signed (positive forward), 0 if unsigned.
Reserved	1-7		*Reserved for future use.*

3.6.6 RawVehicleSpeedOutput (11126)

Message ID: 11126

Protocol Version: 2
Message Version: 0

Description: This message is an output from the device that contains the along-track speed of the vehicle (forward/backward). These measurements come directly from the sensor, and do not have any corrections or calibration applied.

Field	Data Type	Description
Measurement Details	`MeasurementDetails (20B)`	The time of applicability of the measurement data.
Vehicle Speed	`i32`	The vehicle speed estimate (in `m/s * 2e-10`). Set to `0x7fffffff` if unavailable.
Gear	`GearType (u8)`	The transmission gear currently in use (0 if unavailable).
Flags	`u8`	Bitmask of flags associated with measurement data.
Reserved	`u8[2]`	*Reserved for future use.*

Flag bit definitions:

Flag	Bit	Value	Description
`SIGNED`	0	0x01	Set to 1 if the speed value is signed (positive forward), 0 if unsigned.
Reserved	1-7		*Reserved for future use.*

3.6.7 InputDataWrapper (13120)

Message ID: 13120

Protocol Version: 2
Message Version: 0

Description: This message is used to record incoming binary data from sensors, RTK corrections streams, etc. in their native formats for diagnostic purposes.

The size of the content contained in this message, N, must be inferred from the payload length field in the 2.3 Message Header as follows:

```
content_length_bytes = payload_length_bytes -
sizeof(InputDataWrapper)
```

Field	Data Type	Description
System Time	`u40`	The host device timestamp at which the data was captured (in 1e-2 seconds). This timestamp is not aligned with P1 time.
Reserved	`u8[1]`	*Reserved for future use.*
Data Type	`u16`	An identifier for the type of data contained in this message.
Flags	`u8`	Bitmask of flags associated with measurement data.
Reserved	`u8[2]`	*Reserved for future use.*
Content	*Variable (*N)	The captured binary data.

3.7 Device Status Messages

Messages for indicating high-level device status (notifications, software version, etc.).

3.7.1 SystemStatus (10500)

Message ID: 10500

Protocol Version: 2
Message Version: 0

Description: This message describes the overall system status of the device running FusionEngine.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the status message, expressed in the P1 time base.
GNSS Temperature	`i16`	The temperature of the GNSS receiver in Celsius * 2^-7. Set to 0x7FFF if invalid.
Reserved	`u8[118]`	*Reserved for future use.*

3.7.2 SSRStatus (10501)

Message ID: 10501

Protocol Version: 2
Message Version: 0

Description: This message contains the status of the GNSS State Space Representation (SSR) corrections data decoding and modeling process.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the status message, expressed in the P1 time base.
Output GPS Time	`Timestamp (8B)`	The GPS time corresponding with the most recently output corrections data.
Output Source	`RTKOutputSource (u8)`	The source of the most recent RTCM corrections data.
Reserved	`u8`	*Reserved for future use.*
Station ID	`u16`	The RTCM base station ID contained in the most recent corrections data. Set to 0xFFFF if invalid.
Base Latitude	`f64`	The base station latitude (in degrees).
Base Longitude	`f64`	The base station longitude (in degrees).
Base Altitude	`f64`	The base station altitude (in meters).
# Satellites	`u8`	The number of satellites present in the most recent corrections data.
# Signals	`u8`	The number of GNSS signals present in the most recent corrections data.
GNSS Systems	`SatelliteTypeMask (u16)`	A bitmask indicating which GNSS constellations are present in the generated corrections data.
GPS Signal Types	`GPSSignalTypeMask (u16)`	A bitmask indicating which GPS signal types are present in the generated corrections data.
GLO Signal Types	`GLOSignalTypeMask (u16)`	A bitmask indicating which GLONASS signal types are present in the generated corrections data.
GAL Signal Types	`GALSignalTypeMask (u16)`	A bitmask indicating which Galileo signal types are present in the generated corrections data.

BDS Signal Types	`BDSSignalTypeMask (u16)`	A bitmask indicating which BeiDou signal types are present in the generated corrections data.
Reserved	`u8[8]`	*Reserved for future use.*
# GPS Ephemeris	`u8`	The number of GPS satellites for which ephemeris data is available.
# GLO Ephemeris	`u8`	The number of GLONASS satellites for which ephemeris data is available.
# GAL Ephemeris	`u8`	The number of Galileo satellites for which ephemeris data is available.
# BDS Ephemeris	`u8`	The number of BeiDou satellites for which ephemeris data is available.
Reserved	`u8[4]`	*Reserved for future use.*
OSR Status	`u16`	A bitmask indicating the status of OSR corrections data from an external base station.
Reserved	`u8[2]`	*Reserved for future use.*
SSR Status	`u16`	A bitmask indicating the status of incoming SSR corrections data.
Reserved	`u8[1]`	*Reserved for future use.*
SSR Grid ID	`u8`	The ID of the local SSR corrections grid currently in use. Set to 0xFF if invalid.
SSR Enabled Components	`SSRComponentMask (u16)`	A bitmask indicating which SSR model components are enabled for the current region.
SSR Model Status	`SSRComponentMask (u16)`	A bitmask indicating the status of individual SSR model components: 0 = data not available/expired 1 = model data usable
SSR Decode Status	`SSRComponentMask (u16)`	A bitmask indicating the decoding status of incoming SSR data messages: 0 = waiting for data 1 = data received
Reserved	`u8[2]`	*Reserved for future use.*

`RTKOutputSource (u8)`:

Name	Value	Description
None	`0x0`	No RTCM output available.

OSR	`0x1`	RTCM output received from an incoming RTK base station.
SSR	`0x2`	RTCM output generated using SSR model data.

`RTKOutputSource (u8)`:

Name	Value	Description
None	`0x0`	No RTCM output available.
OSR	`0x1`	RTCM output received from an incoming RTK base station.
SSR	`0x2`	RTCM output generated using SSR model data.

OSR base station data status bitmask (`u16`):

Bit Number	Value	Description
0	`0x0001`	OSR corrections data available
1-15		*Reserved for future use.*

SSR model data status bitmask (`u16`):

Bit Number	Value	Description
0	`0x0001`	SSR model data ready for OSR generation
1-15		*Reserved for future use.*

`SSRComponentMask (u16)`:

Bit Number	Value	Description
0	`0x0001`	SSR network metadata
1	`0x0002`	Grid definition
2	`0x0004`	Satellite group definition
3	`0x0008`	Geoid model data
4	`0x0010`	Antenna corrections data (ATX)
5	`0x0020`	High-rate satellite corrections data
6	`0x0040`	Low-rate satellite corrections data
7	`0x0080`	Global per-satellite ionosphere data (GSI)
8	`0x0100`	Gridded per-satellite ionosphere data (GRI)
9	`0x0200`	Gridded troposphere data (GRT)
10	`0x0400`	Regional per-satellite ionosphere data (RSI)
11	`0x0800`	Global vertical ionosphere data (GVI)
12	`0x1000`	Regional troposphere data (RT)
13-15		*Reserved for future use.*

3.7.3 VersionInfoMessage (13003)

Message ID: 13003

Protocol Version: 2
Message Version: 0

Description: Describes the version of the system and its subcomponents.

Note that the strings contained in this message are **not** null-terminated.

Field	Data Type	Description
System Time	`i64`	Current system time (in nanoseconds).
Firmware Version Length	`u8`	The length of the firmware string (in bytes).
Fusion Engine Version Length	`u8`	The length of the Fusion Engine string (in bytes).
Operating System (OS) Version Length	`u8`	The length of the operating system string (in bytes).
Receiver Version Length	`u8`	The length of the receiver string (in bytes).
Reserved	`u8[4]`	*Reserved for future use.*
Firmware String Description	`str`	ASCII string representing the firmware version.
Engine String Description	`str`	ASCII string representing the navigation software version (FusionEngine).
Operating System (OS) String Description	`str`	ASCII string representing the operating system, kernel, and/or bootloader version.
Receiver String Description	`str`	ASCII string representing the software version of the underlying measurement engine.
<Padding>	`u8`	This message will be padded to a length that is a multiple of 4 bytes.

3.7.4 EventNotificationMessage (13004)

Message ID: 13004

Protocol Version: 2
Message Version: 0

Description: Notification of a system event for logging purposes.

Note that the string contained in this message is **not** null-terminated.

Field	Data Type	Description
Event Type	`EventType (u8)`	The type of event that occurred.
Reserved	`u8[3]`	*Reserved for future use.*
System Time	`i64`	The system time when the event occurred (in nanoseconds).
Flags	`u64`	Additional flags or information associated with the event (if applicable).
Description Length	`u16`	The length of the event description string (in bytes).
Description	`str`	ASCII string description of the event. Not null-terminated.

`EventType (u8):`

Event Type	Value	Description
`LOG`	`0x0`	Event containing a logged message string from the device.
`RESET`	`0x1`	Event indicating a device reset occurred. The event flags will be set to the requested reset bitmask. The payload will contain a string describing the cause.
`CONFIG_CHANGE`	`0x2`	Notification that the user configuration has been changed.
`COMMAND`	`0x3`	Notification that the user performed a command (e.g., configuration request, fault injection enable/disable).
`COMMAND_RESPONSE`	`0x4`	Record containing the response to a user command. Response events are not output on the interface on which the command was received; that interface will receive the response itself.

3.7.5 DeviceIDMessage (13007)

Message ID: 13007

Protocol Version: 2
Message Version: 0

*Description: Contains various device identifiers in the host device. Note: The strings are **not** null-terminated. Note that the ID fields may contain either a string (not null-terminated) or a binary value, depending on the type of device.*

Field	Data Type	Description
System Time	`i64`	System Time in nanoseconds.
Device Type	`DeviceType (u8)`	The type of device responding.
HW ID Length	`u8`	Length of the hardware ID field (in bytes).
User ID Length	`u8`	Length of the User ID field (in bytes).
GNSS ID Length	`u8`	Length of the GNSS ID field (in bytes).
Reserved	`u8[4]`	*Reserved for future use.*
HW ID	`u8[x]`	The value of the hardware ID. The length of this string is defined by the `HW ID Length` parameter.
User ID	`u8[y]`	The value of the User ID. The length of this string is defined by the `User ID Length` parameter.
GNSS ID	`u8[z]`	The value of the GNSS ID. The length of this string is defined by the `GNSS ID Length` parameter.

`DeviceType (u8)`:

Response Type	Response Code
Unknown	0
Point One Atlas	1
Quectel LG69T-AM	2
Quectel LG69T-AP	3
Quectel LG69T-AH	4
Nexar Beam2K	5
Point One Embedded SSR Client	6
Point One Desktop SSR Client	7

3.8 GNSS Corrections Messages

Messages containing GNSS corrections data or details.

3.8.1 LBandFrameMessage

Message ID: 14000

Protocol Version: 2
Message Version: 0

Description: L-band user data frame contents.

Field	Data Type	Description
System Time	`i64`	The system time when the frame was received in nanoseconds. Note: this time is not synchronized to other P1 systems or GNSS.
User Data Size	`u16`	Number of bytes in this data payload.
Bit Error Count	`u16`	Count of bit errors found in the data frame.
Signal Power	`u8`	Power of the signal in decibels.
Reserved	`u8[3]`	*Reserved for future use.*
Doppler Offset	`f32`	The offset from the center frequency in hertz. This includes effects from user motion, receiver clock, and satellite clock errors.
Value	*Variable*	The beginning of the demodulated L-band frame data.

3.9 ROS Support

The messages defined in this file are intended to be translated into their corresponding ROS message structures where ROS integration is needed.

Note: The messages defined here are _not_ guaranteed to be byte-compatible with ROS messages. They are designed to have the same or similar content, which can be easily copied into a ROS message.

3.9.1 ROSPose (12000)

Message ID: 12000

Protocol Version: 2
Message Version: 0

Description: Position, velocity, and attitude (orientation) solution, in ROS Pose message format.

Note that this message is designed to be easily convertible to a ROS Pose message, but cannot be directly cast into a ROS Pose object.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
East Position	`f64`	The position offset in the East direction relative to the first valid ROS Pose (in meters).
North Position	`f64`	The position offset in the North direction relative to the first valid ROS Pose (in meters).
Up Position	`f64`	The position offset in the Up direction relative to the first valid ROS Pose (in meters).
Attitude Quaternion	`f64[4]`	The platform body orientation with respect to the local ENU frame, represented as a quaternion with the scale component last (x, y, z, w).

3.9.2 ROSGPSFix (12010)

Message ID: 12010

Protocol Version: 2
Message Version: 0

Description: Information about the position, altitude, and error in the ROS GPSFix message format.

Note that this message is designed to be easily convertible to a ROS GPSFix message, but cannot be directly cast into a ROS GPSFix object.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
Latitude	`f64`	The geodetic latitude in degrees, expressed using the WGS-84 reference ellipsoid.
Longitude	`f64`	The geodetic longitude in degrees, expressed using the WGS-84 reference ellipsoid.
Altitude	`f64`	The geodetic altitude (height above the ellipsoid) in meters, expressed using the WGS-84 reference ellipsoid.
Track Angle	`f64`	The vehicle direction from north in degrees.
Vehicle Speed	`f64`	The vehicle ground speed in meters/second.
Climb Speed	`f64`	The vehicle vertical speed in meters/second.
Pitch	`f64`	*Not current supported.*
Roll	`f64`	*Not current supported.*
Dip	`f64`	*Not current supported.*
GPS Time	`Timestamp (8B)`	The GPS time of the solution, if available.
GDOP	`f64`	The geometric dilution of precision (GDOP).
PDOP	`f64`	The position dilution of precision (PDOP).
HDOP	`f64`	The horizontal dilution of precision (HDOP).
VDOP	`f64`	The vertical dilution of precision (VDOP).
TDOP	`f64`	The time dilution of precision (TDOP).
3D Error	`f64`	The spherical position uncertainty in meters.
Horizontal Error	`f64`	The horizontal position uncertainty in meters.
Vertical Error	`f64`	The vertical position uncertainty in meters.
Track Error	`f64`	The track uncertainty in meters.
Vehicle Speed Error	`f64`	The vehicle ground speed uncertainty in meters/second
Climb Speed Error	`f64`	The vehicle vertical speed uncertainty in meters/second.

Time Error	`f64`	The time uncertainty in seconds.
Pitch Error	`f64`	The platform pitch uncertainty in degrees.
Roll Error	`f64`	The platform roll uncertainty in degrees.
Dip Error	`f64`	The platform dip uncertainty in degrees.
Position Covariance	`f64[9]`	The 3x3 position covariance matrix (in meters^2), resolved in the local ENU frame and stored in row-major order.
Position Covariance Type	`u8`	The method in which position covariance was populated (see below).
Reserved	`u8[3]`	*Reserved for future use.*

`CovarianceType (u8):`

Type	Value	Description
Unknown	`0x0`	Position covariance is unknown.
Approximated	`0x1`	Position covariance entries are approximate.
Diagonal Known	`0x2`	Position covariance diagonal entries are known.
Known	`0x3`	Position covariance is known.
Reserved	`0x4-0xFF`	*Reserved.*

3.9.3 ROSIMUMessage (12011)

Message ID: 12011

Protocol Version: 2
Message Version: 0

Description: Corrected IMU measurement data in the ROS IMU message format. This data is an output from the system. It cannot be used to send IMU measurements to a FusionEngine device.

Note that this message is designed to be easily convertible to a ROS IMU message, but cannot be directly cast into a ROS IMU object.

The data contained in this message has been corrected for estimated accelerometer and gyro errors, including biases and scale factors, and has been rotated into the vehicle body frame from the original IMU orientation.

See http://docs.ros.org/api/sensor_msgs/html/msg/Imu.html for the related ROS message definition.

Field	Data Type	Description
P1 Time	`Timestamp (8B)`	The time of the solution, expressed in the P1 time base.
Orientation	`f64[4]`	The platform body orientation with respect to the local ENU frame, represented as a quaternion with the scale component last (x, y, z, w).
Orientation Covariance	`f64[9]`	The orientation covariance in row-major format about the X, Y, Z axes. Per the ROS IMU message specification: - If the a value is known but its covariance is not, its covariance matrix will be set to 0.0 - If a value is not known or not available, its covariance matrix will be set to -1.0 - The value itself will be set to `NAN`, as this is not specified in the ROS message definition Note that the ROS IMU message does not use `NAN` in the covariance matrix to represent either data or covariance not known.
X Rotation Rate	`f64`	The vehicle X axis gyroscope rate of rotation in radians/second.

Y Rotation Rate	`f64`	The vehicle Y axis gyroscope rate of rotation in radians/second.
Z Rotation Rate	`f64`	The vehicle Z axis gyroscope rate of rotation in radians/second.
Angular Velocity Covariance	`f64[9]`	The vehicle X/Y/Z gyroscope rate of rotation covariance matrix. Each value set to -1.0 if not available.
X Acceleration	`f64`	The vehicle X acceleration in meters/second^2.
Y Acceleration	`f64`	The vehicle Y acceleration in meters/second^2.
Z Acceleration	`f64`	The vehicle Z acceleration in meters/second^2.
Acceleration Covariance	`f64[9]`	The vehicle X/Y/Z acceleration covariance matrix. Each value set to -1.0 if not available.

3.10 Platform-Specific Messages

This section describes proprietary messages that are designed for use with specific devices, platforms, or hardware, and do not apply to all FusionEngine devices.

3.10.1 STA5635Command (14100)

Message ID: 14100

Protocol Version: 2
Message Version: 0

Description: Send a command to an attached STMicroelectronics STA5635 RF front-end.

See the STA5635 data sheet for available command, address, and data values.

Field	Data Type	Description
Command	`u8`	The STA5635 command code to be issued.
Address	`u8`	The address of the STA5635 register to be accessed.
Data	`u8[2]`	The value to be sent to the device, where data[0] contains the MSB.

3.10.2 STA5635CommandResponse (14101)

Message ID: 14101

Protocol Version: 2
Message Version: 0

Description: Result from an STMicroelectronics STA5635 RF front-end sent in response to a 3.10.1 STA5635Command (14100).

Field	Data Type	Description
System Time	`i64`	The system time when the response was received (in nanoseconds).
Sequence Number	`u32`	The address of the STA5635 register to be accessed.
Data	`u8[4]`	The response from the device, where data[0] contains the first byte in the response.

3.10.3 STA5635IQData (14102)

Message ID: 14102

Protocol Version: 2
Message Version: 0

Description: IQ sample data captured by an STMicroelectronics STA5635 RF front-end.

The size of the IQ contents contained in this message, N, must be inferred from the payload length field in the 2.3 Message Header as follows:

```
content_length_bytes = payload_length_bytes -
sizeof(InputDataWrapper)
```

Field	Data Type	Description
Reserved	`u8[4]`	*Reserved for future use.*
Sample Data	`N`	The captured IQ sample data.

4 Proprietary NMEA Sentence Definitions

Point One supports a limited set of standard and proprietary NMEA-0183 messages. These messages offer a limited subset of the functionality of the FusionEngine binary protocol.

4.1 $P1CALSTATUS

Description: Indicates the status of the internal calibration algorithm.

See also 3.3.5 CalibrationStatus (10004).

Sentence Structure

```
$P1CALSTATUS,<stage>,<state_verified>,<gyro_percent_complete>,<accel_
percent_complete>,<mounting_angle_percent_complete>
```

Field	Description
stage	Represents the current calibration stage • 0 - Unknown/not started • 1 - Initial mounting angle convergence (performance may be slightly degraded) • 2 - Final mounting angle convergence • 255 - Done
state_verified	• 0 - Calibration is waiting for the navigation state to be verified. • 1 - The navigation state has been verified after a hot start and calibration can proceed
gyro_percent_complete	Gyroscope correction calibration status percentage
accel_percent_complete	Accelerometer correction calibration status percentage
mounting_angle_percent_complete	IMU mounting angle correction completion percentage

4.2 $P1MSG

Description: Notification of a system event for logging purposes.

See also 3.7.4 EventNotificationMessage (13004).

Sentence Structure

```
$P1MSG,<system_time_ns>,<severity>,<message>
```

Field	Description
system_time_ns	The system/OS time when the message was generated.
severity	A number indicating the severity of the message. Lower numbers are more severe.
message	An ASCII string containing the message.

4.3 $PQTMGNSS

Description: Quectel proprietary message to output information regarding GNSS.

Sentence Structure

```
$PQTMGNSS,<TBD>
```

Field	Description
TBD	TBD

4.4 $PQTMVER

Description: Quectel proprietary message to indicate the version of the software.

See also 3.7.3 VersionInfoMessage (13003).

```
Sentence Structure
$PQTMVER,<TBD>
```

Field	Description
TBD	TBD

4.5 $PQTMVERNO

Description: Quectel proprietary message to indicate the version of the software.

See also 3.7.3 VersionInfoMessage (13003).

Sentence Structure

```
$PQTMVERNO,<TBD>
```

Field	Description
TBD	TBD

Appendix A. Document Version History

Version 0.22 (2025-3-17)

Compatibility with FusionEngine Protocol Release v1.24.0

- Added a section describing the vehicle body axis and attitude angle definitions
- Added GNSSAttitudeOutput and RawGNSSAttitudeOutput message definitions, and removed deprecated HeadingOutput and RawHeadingOutput message definitions
- Deprecated HEADING_BIAS config type in favor of new `GNSS_AUX_LEVER_ARM` parameter
- Added flags field to PlatformStorageMessage to indicate user configuration data source when available
- Added type definition for NMEA-0183 ZDA messages
- Added `RESET_GNSS_TIME` bit definition
- Added command Response enum value for for an unsupported communication interface request
- Added flags bitmask to PoseMessage
- Added InputDataWrapper message definition
- Added STA5635 control and IQ data messages

Version 0.21 (2024-8-19)

Compatibility with FusionEngine Protocol Release v1.23.4

- Added IMUInput message definition
- Added StartupRequest message definition
- Added LBandFrame and SSRStatus message definitions
- Added ImportData, ExportData, and PlatformStorageData message definitions
- Added I/O interface ENABLED configuration parameter
- Added UNIX domain socket I/O interface support
- Added ConfigurationSource option to query default settings
- Added InterfaceConfigType definitions for enable/disable and transport direction
- Added UNIX domain socket support
- Added ionosphere model SBAS option
- Changed LBandConfig center frequency to f64, and added additional PMP configuration settings for data rate and descrambler control
- Renamed RESET_GNSS_MEASUREMENT_ENGINE ResetRequest type to REBOOT_GNSS_MEASUREMENT_ENGINE
- Added additional NMEA message types
- Added user device ID and L-band enumerations under ConfigType list
- Added additional vehicle models

- Updated SetMessageRate and GetMessageRate message definitions
- Added ResponseCode enumerations

Version 0.20 (2023-11-10)

Compatibility with FusionEngine Protocol Release v1.22.3

- Updates for wheel encoder messages
- Fixed data types for wheel encoder and velocity inputs
- Added messages for filtered velocity and wheel encoder outputs

Version 0.19 (2023-10-05)

Compatibility with FusionEngine Protocol Release v1.22.0

- Added new SupportedIOInterfaces message
- Corrected VersionInformation documentation
- Updated WheelSensorType enum values
- Correction protocol version number

Version 0.18 (2023-8-24)

Compatibility with FusionEngine Protocol Release v1.21.0

- Updated Fault Control types to include factory testing options

Version 0.17 (2023-6-9)

Compatibility with FusionEngine Protocol Release v1.18.1

- Updated ROS IMU message documentation
- Adds Heading Bias configuration
- Changed HeadingMeasurement to RawHeadingOutput (no change to struct)
- Added HeadingOutput message
- Added DeviceID message

Version 0.16 (2023-5-4)

Compatibility with FusionEngine Protocol Release v1.18.1

- Added configuration items for the troposphere and ionosphere.

Version 0.15 (2023-4-7)

Compatibility with FusionEngine Protocol Release v1.18.0

- Added leap second and GPS week rollover override configuration options

- Added UART1 Baud Rate configuration option
- Revised fields in the GNSSInfo Message to include UTC leap seconds, number of SVs, GNSS Correction age and GNSS baseline distance.
- Added MeasurementDetails definition
- Added Sensor Data Source definition for measurement inputs and outputs.
- Removed MeasurementTimestamp definition
- Removed deprecated WheelSpeed and VehicleSpeed messages
- Added RawIMUOutput, RawWheelTickOutput, RawVehicleTickOutput, RawWheelSpeedOutput, RawVehicleSpeedOutput
- Added RawWheelTickInput, RawVehicleTickInput, RawWheelSpeedInput, RawVehicleSpeedInput
- Added SystemStatus message
- Added new Diagnostic Reset functionality and modified HOT, WARM and COLD start mask definitions.
- Added Revert To Default option for configuration settings.
- Clarified definition of SaveAction in SetConfig message.

Version 0.14 (2023-3-3)

Compatibility with FusionEngine Protocol Release v1.17.0

- Added GNSS system and frequency band enable/disable settings definitions
- Clarified use of WheelConfig (software data/CAN) vs HardwareTickConfig (voltage signal)

Version 0.13 (2023-2-17)

Compatibility with FusionEngine Protocol Release v1.15.0

- Added HeadingMeasurement message definition

Version 0.12 (2022-12-6)

Compatibility with FusionEngine Protocol Release v1.13.0

- Clarified payload descriptions for SetConfig, ConfigResponse, and FaultControl messages
- Adds ROS compatible messages for ROSPose, ROSGPSFix and ROSImuMessage

Version 0.11 (2022-11-7)

Compatibility with FusionEngine Protocol Release v1.13.0

- Corrected missing reserved bytes in FaultControl definition
- Added EventNotification message definition

Version 0.10 (2022-11-4)

Compatibility with FusionEngine Protocol Release v1.13.0

- Added ALL and CURRENT TransportType enumerations
- Corrected UNSUPPORTED_VERSION response code documentation
- Added FaultControl message definition

Version 0.9 (2022-10-10)

Compatibility with FusionEngine Protocol Release v1.12.0

- Added apply-and-save flag to SetMessageRate command
- Added wildcard capabilities to SetMessageRate and GetMessageRate commands
- Removed SetOutputInterfaceConfig and GetOutputInterfaceConfig commands

Version 0.8 (2022-7-29)

Compatibility with FusionEngine Protocol Release v1.11.2

- Added watchdog enable/disable setting definition
- Added message type enumeration for NMEA-0183

Version 0.7 (2022-7-22)

Compatibility with FusionEngine Protocol Release v1.11.0

- Added Vehicle Tick Measurement definition
- Added control messages for enabling/disabling/controlling rate of individual output message types
- Changed inbound measurement timestamp definitions (MeasurementTimestamps)

Version 0.6 (2022-7-15)

Compatibility with FusionEngine Protocol Release v1.11.0

- Added WheelSpeedMeasurement, WheelTickMeasurement, and VehicleSpeedMeasurement message definitions
- Fixed missing reserved bytes in GetOutputInterfaceConfig

Version 0.5 (2022-7-11)

Compatibility with FusionEngine Protocol Release v1.10.0

- Corrected descriptions for WheelConfig, HardwareTickConfig,
- Fixed reserved bytes in SetConfig and OutputInterfaceConfigurationEntry

Version 0.4 (2022-6-22)

Compatibility with FusionEngine Protocol Release v1.10.0

- Added HardwareTickConfig configuration payload
- Compatibility with FusionEngine Release v1.10.0

Version 0.3 (2022-5-31)

Compatibility with FusionEngine Protocol Release v1.10.0

- Added RelativeENUPosition and ShutdownRequest messages
- Added reboot support to ResetRequest message

Version 0.2 (2022-5-9)

Compatibility with FusionEngine Protocol Release v1.10.0

- Added ResetRequest, SetConfig, GetConfig, and ConfigResponse message definitions.
- Added solution and measurement message definitions
  (Pose, PoseAux, IMUMeasurement, etc.).

Version 0.1 (2022-2-18)

Compatibility with FusionEngine Protocol Release v1.9.0

- Initial version.