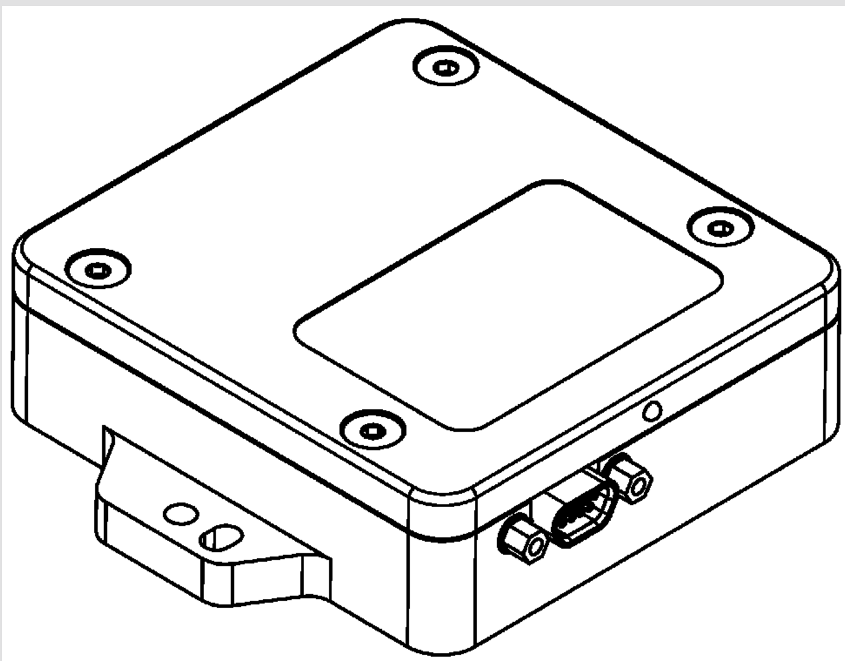
LORD MANUAL

3DM-GQ4™-45

Data Communications Protocol



LORD MicroStrain®
SENSING SYSTEMS

MicroStrain® Sensing Systems
459 Hurricane Lane
Suite 102
Williston, VT 05495
United States of America
Phone: 802-862-6629
Fax: 802-863-4093

www.microstrain.com
sensing_support@LORD.com

REVISED: September 26, 2015

Table of Contents

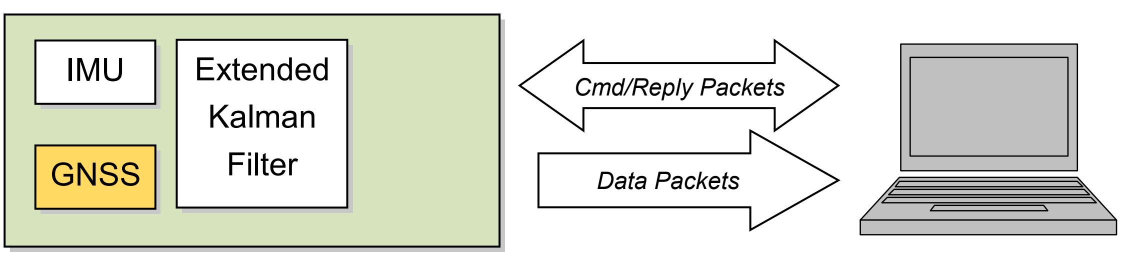
3DM-GQ4-45 API

API Introduction

The 3DM-GQ4-45 programming interface is comprised of a compact set of setup and control commands and a very flexible user-configurable data output format. The commands and data are divided into 4 command sets and 3 data sets corresponding to the internal architecture of the device. The four command sets consist of a set of "Base" commands (a set that is common across many types of devices), a set of unified "3DM" (3D Motion) commands that are specific to the MicroStrain inertial product line, a set of "Estimation Filter" commands that are specific to MicroStrain navigation and advanced AHRS devices, and a set of "System" commands that are specific to sensor systems comprised of more than one internal sensor block. The three data sets represent the three types of data that the GQ4-45 is capable of producing: "IMU" (Inertial Measurement Unit) data, "GNSS" (Global Navigation Satellite System) data, and "Estimation Filter" (Position, Velocity, and Attitude) data. The type of estimation filter used in the GQ4-45 is an Extended Kalman Filter (EKF).

Base commands	*Ping, Idle, Resume, Get ID Strings, etc.*
3DM commands	*Poll IMU Data, Poll GNSS Data, etc.*
Estimation Filter commands	*Reset Filter, Sensor to Vehicle Frame Transformation, etc.*
System commands	*Switch Communications Mode, etc.*
IMU data	*Acceleration Vector, Gyro Vector, etc.*
GNSS data	*Latitude, Longitude, UTC, Satellites in view, etc.*
Estimation Filter data	*Position, Velocity, Attitude, Acceleration Estimates, etc.*

The protocol is packet based. All commands, replies, and data are sent and received as fields in a message packet. Commands are all confirmed with an ack/nack (with a few exceptions). The packets have a descriptor type field based on their contents, so it is easy to identify if a packet contains commands, replies, IMU data, GNSS data, or Estimation Filter data.



The 3DM-GQ4-45 has an advanced mode switch that allows the device to switch into direct "Sensor" or "GNSS" mode. In those modes, the device responds to the native protocols of the 3DM-GQ4-45 IMU or the u-blox M8M GNSS devices which are embedded in the 3DM-GQ4-45. These modes can be used to access advanced or specialized features of these devices (see the Advanced Programming section).

Command and Data Summary

Below is a summary of the commands and data available in the programming interface. Commands and data are denoted by two values. The first value denotes the "descriptor set" that the command or data belongs to (Base command, 3DM command, Estimation Filter Command, IMU data, GNSS data, or Estimation Filter data) and the second value denotes the unique command or data "descriptor" in that set. The pair of values constitutes a "full descriptor".

Commands

Base Command Set (0x01)

- Ping (0x01, 0x01)
- Set To Idle (0x01, 0x02)
- Get Device Information (0x01, 0x03)
- Get Device Descriptor Sets (0x01, 0x04)
- Device Built-In Test (BIT) (0x01, 0x05)
- Resume (0x01, 0x06)
- GPS Time Update (0x01, 0x72)
- Device Reset (0x01, 0x7E)

3DM Command Set (0x0C)

- Poll IMU Data (0x0C, 0x01)
- Poll GNSS Data (0x0C, 0x02)
- Poll Estimation Filter Data (0x0C, 0x03)
- Get IMU Data Rate Base (0x0C, 0x06)
- Get GNSS Data Rate Base (0x0C, 0x07)
- Get Estimation Filter Data Rate Base (0x0C, 0x0B)
- IMU Message Format (0x0C, 0x08)
- GNSS Message Format (0x0C, 0x09)
- Estimation Filter Message Format (0x0C, 0x0A)
- GNSS Constellation Settings (0x0C, 0x21)
- GNSS SBAS Settings (0x0C, 0x22)
- Enable/Disable Device Continuous Data Stream (0x0C, 0x11)
- Device Startup Settings (0x0C, 0x30)
- Accel Bias (0x0C, 0x37)
- Gyro Bias (0x0C, 0x38)
- Capture Gyro Bias (0x0C, 0x39)
- Magnetometer Hard Iron Offset (0x0C, 0x3A)
- Magnetometer Soft Iron Matrix (0x0C, 0x3B)
- Coning and Sculling Enable (0x0C, 0x3E)
- Change UART BAUD rate (0x0C, 0x40)
- Advanced Low-Pass Filter Settings (0x0C, 0x50)
- Complementary Filter Settings (0x0C, 0x51)
- Device Status* (0x0C, 0x64)

Estimation Filter Command Set (0x0D)

System Command Set (0x7F)

**Advanced Commands*

Data

IMU Data Set (set 0x80)

- Scaled Accelerometer Vector (0x80, 0x04)
- Scaled Gyro Vector (0x80, 0x05)
- Scaled Magnetometer Vector (0x80, 0x06)
- Scaled Ambient Pressure (0x80, 0x17)
- Delta Theta Vector (0x80, 0x07)
- Delta Velocity Vector (0x80, 0x08)
- CF Orientation Matrix (0x80, 0x09)
- CF Quaternion (0x80, 0x0A)
- CF Euler Angles (0x80, 0x0C)
- CF Stabilized Mag Vector (North) (0x80, 0x10)
- CF Stabilized Accel Vector (Up) (0x80, 0x11)
- GPS Correlation Timestamp (0x80, 0x12)

GNSS Data Set (set 0x81)

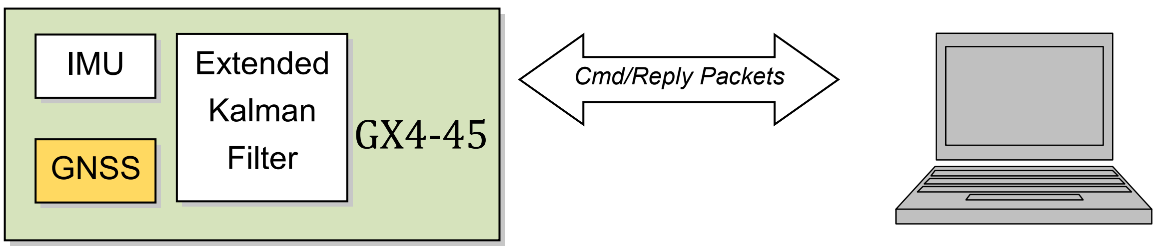
- LLH Position (0x81, 0x03)
- ECEF Position (0x81, 0x04)
- NED Velocity (0x81, 0x05)
- ECEF Velocity (0x81, 0x06)
- Dilution of Precision (DOP) (0x81, 0x07)
- UTC Time (0x81, 0x08)
- GPS Time (0x81, 0x09)
- Clock Information (0x81, 0x0A)
- GNSS Fix Information (0x81, 0x0B)
- Space-Vehicle Information (SVI) (0x81, 0x0C)
- Hardware Status (0x81, 0x0D)
- DGNSS Information (0x81, 0x0E)
- DGNSS Channel Status (0x81, 0x0F)

Estimation Filter Data Set (set 0x82)

- Filter Status (0x82, 0x10)
- GPS Timestamp (0x82, 0x11)
- LLH Position (0x82, 0x01)
- NED Velocity (0x82, 0x02)
- Orientation, Quaternion (0x82, 0x03)
- Orientation, Matrix (0x82, 0x04)
- Orientation, Euler Angles (0x82, 0x05)
- Gyro Bias (0x82, 0x06)
- Accel Bias (0x82, 0x07)
- LLH Position Uncertainty (0x82, 0x08)
- NED Velocity Uncertainty (0x82, 0x09)
- Attitude Uncertainty, Euler Angles (0x82, 0x0A)
- Gyro Bias Uncertainty (0x82, 0x0B)

Basic Programming

The 3DM-GQ4-45 is designed to stream IMU, and GNSS, and Estimation Filter data packets over a common interface as efficiently as possible. To this end, programming the device consists of a configuration stage where the data messages and data rates are configured. The configuration stage is followed by a data streaming stage where the program starts the incoming data packet stream.



In this section there is an overview of the packet, an overview of command and reply packets, an overview of how an incoming data packet is constructed, and then an example setup command sequence that can be used directly with the 3DM-GQ4-45 either through a COM utility or as a template for software development.

MIP Packet Overview

This is an overview of the 3DM-GQ4-45 packet structure. The packet structure used is the MicroStrain "MIP" packet. A reference to the general packet structure is presented in the MIP Packet Reference section. An overview of the packet is presented here.

The MIP packet "wrapper" consists of a four byte header and two byte checksum footer:

Header				Packet Payload			Checksum	
SYNC1 "u"	*SYNC2 "e"*	*Descriptor Set byte*	*Payload Length byte*	*Field Length byte*	*Field Descriptor byte*	*Field Data*	*MSB*	*LSB*
0x75	**0x65**	**0x80**	**0x0E**	**0x0E**	**0x03**	**0x3E 7A 63 A0 0xBB 8E 3B 29 0x7F E5 BF 7F**	**0x83**	**0xE1**

Payload Length byte. This specifies the length of the packet payload. The packet payload may contain one or more fields and thus this byte also represents the sum of the lengths of all the fields in the payload.

Descriptor Set. Descriptors are grouped into different sets. The value 0x80 identifies this packet as an AHRS data packet. Fields in this packet will be from the AHRS data descriptor set.

Start of Packet (SOP) "sync" bytes. These are the same for every MIP packet and are used to identify the start of the packet.

2 byte Fletcher checksum of all the bytes in the packet.

The packet payload section contains one or more fields. Fields have a length byte, descriptor byte, and data. The diagram below shows a packet payload with a single field.

Header				Packet Payload			Checksum	
SYNC1 "u"	*SYNC2 "e"*	*Descriptor Set byte*	*Payload Length byte*	*Field Length byte*	*Field Descriptor byte*	*Field Data*	*MSB*	*LSB*
0x75	0x65	0x80	0x0E	**0x0E**	**0x06**	**0x3E 7A 63 A0 0xBB 8E 3B 29 0x7F E5 BF 7F**	0x86	0x08

Field Length byte. This represents a count of all the bytes in the field including the length byte, descriptor byte and field data.

Descriptor byte. This byte identifies the contents of the field data. This descriptor indicates that the data is a mag vector (set: 0x80, descriptor: 0x06)

Field data. The length of the data is Field Length – 2. This data is 12 bytes long (14 – 2) and represents the floating point magnetometer vector value from the AHRS data set.

Below is an example of a packet payload with two fields (gyro vector and mag vector). Note the payload length byte of 0x1C which is the sum of the two field length bytes 0x0E + 0x0E:

Header				Packet Payload (2 fields)						Checksum	
SYNC1 "u"	*SYNC2 "e"*	*Descriptor Set*	*Payload Length*	*Field1 Len*	*Field1 Descriptor*	*Field1 Data*	*Field2 Len*	*Field2 Descriptor*	*Field2 Data*	*MSB*	*LSB*
0x75	0x65	0x80	**0x1C**	**0x0E**	0x05	0x3E 7A 63 A0 0xBB 8E 3B 29 0x7F E5 BF 7F	**0x0E**	0x06	0x3E 7A 63 A0 0xBB 8E 3B 29 0x7F E5 BF 7F	0xB1	0x1E

Command Overview

The basic command sequence begins with the host sending a command to the device. A command packet contains a field with the command value and any command arguments.

The device responds by sending a reply packet. The reply contains at minimum an ACK/NACK field. If any additional data is included in a reply, it appears as a second field in the packet.

Example "Ping" Command Packet:

Below is an example of a "Ping" command packet from the Base command set. A "Ping" command has no arguments. Its function is to determine if a device is present and responsive:

Header				Packet Payload			Checksum	
SYNC1 "u"	*SYNC2 "e"*	*Descriptor Set byte*	*Payload Length byte*	*Field Length byte*	*Field Descriptor byte*	*Field Data*	*MSB*	*LSB*
0x75	**0x65**	**0x01**	**0x02**	**0x02**	**0x01**	*N/A*	**0xE0**	**0xC6**

Copy-Paste version: "7565 0102 0201 E0C6"

The packet header has the "ue" starting sync bytes characteristic of all MIP packets. The descriptor set byte (0x01) identifies the data as being from the Base command set. The length of the payload portion is 2 bytes. The payload portion of the packet consists of one field. The field starts with the length of the field which is followed by the descriptor byte (0x01) of the field. The field descriptor value *is* the command value. Here the descriptor identifies the command as the "Ping" command from the Base command descriptor set. There are no parameters associated with the ping command, so the field data is empty. The checksum is a two byte Fletcher checksum (see the MIP Packet Reference for instructions on how to compute a Fletcher two byte checksum).

Example "Ping" Reply Packet:

The "Ping" command will generate a reply packet from the device. The reply packet will contain an ACK/NACK field. The ACK/NACK field contains an "echo" of the command byte plus an error code. An error code of 0 is an "ACK" and a non-zero error code is a "NACK":

Header				Packet Payload			Checksum	
SYNC1 "u"	*SYNC2 "e"*	*Descriptor Set byte*	*Payload Length byte*	*Field Length byte*	*Field Descriptor byte*	*Field Data: 2 bytes*	*MSB*	*LSB*
0x75	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Command echo: **0x01** Error code: **0x00**	**0xD5**	**0x6A**

Copy-Paste version: "7565 0104 04F1 0100 D56A"

The packet header has the "ue" starting sync bytes characteristic of all MIP packets. The descriptor set byte (0x01) identifies the payload fields as being from the Base command set. The length of the payload portion is 4 bytes. The payload portion of the packet consists of one field. The field starts with the length of the field which is followed by the descriptor byte (0xF1) of the field. The field descriptor byte identifies the reply as the "ACK/NACK" from the Base command descriptor set. The field data consists of an "echo" of the original command (0x01) followed by the error code for the command (0x00). In this case the error is zero, so the field represents an "ACK". Some examples of non-zero error codes that might be sent are "timeout", "not implemented", and "invalid parameter in command". The checksum is a two byte Fletcher checksum (see the MIP Packet Reference for instructions on how to compute a Fletcher two byte checksum).

The ACK/NACK descriptor value (0xF1) is the same in all descriptor sets. The value belongs to a set of reserved global descriptor values.

The reply packet may have additional fields that contain information in reply to the command. For example, requesting Device Status will result in a reply packet that contains two fields in the packet payload: an ACK/NACK field and a device status information field.

Data Overview

Data packets are generated by the device. When the device is powered up, it may be configured to immediately stream data packets out to the host or it may be "idle" and waiting for a command to either start continuous data or to get data by "polling" (one data packet per request). Either way, the data packet is generated by the device in the same way.

Example Data Packet:

Below is an example of a MIP data packet which has one field that contains the scaled accelerometer vector.

Header				Packet Payload			Checksum	
SYNC1 "u"	*SYNC2 "e"*	*Descriptor Set byte*	*Payload Length byte*	*Field Length byte*	*Field Descriptor byte*	*Field Data: Accel vector (12 bytes, 3 float – X, Y, Z)*	*MSB*	*LSB*
0x75	**0x65**	**0x80**	**0x0E**	**0x0E**	**0x04**	**0x3E 7A 63 A0 0xBB 8E 3B 29 0x7F E5 BF 7F**	**0x92**	**0xC0**

Copy-Paste version: "7565 800E 0E04 3E7A 63A0 BB8E 3B29 7FE5 BF7F 92C0"

The packet header has the "ue" starting sync bytes characteristic of all MIP packets. The descriptor set byte (0x80) identifies the payload field as being from the IMU data set. The length of the packet payload portion is 14 bytes (0x0E). The payload portion of the packet starts with the length of the field. The field descriptor byte (0x04) identifies the field data as the scaled accelerometer vector from the IMU data descriptor set. The field data itself is three single precision floating point values of 4 bytes each (total of 12 bytes) representing the X, Y, and Z axis values of the vector. The checksum is a two byte Fletcher checksum (see the MIP Packet Reference for instructions on how to compute a Fletcher two byte checksum).

The format of the field data is fully and unambiguously specified by the descriptor. In this example, the field descriptor (0x04) specifies that the field data holds an array of three single precision IEEE-754 floating point numbers in big-endian byte order and that the values represent units of "g's" and the order of the values is X, Y, Z vector order. Any other specification would require a different descriptor (see the Data Reference section of this manual).

Each packet can contain any combination of data quantities from the same data descriptor set (any combination of GNSS data OR any combination of IMU data OR and combination of Estimation Filter data– you cannot combine GNSS data, IMU data, and Estimation Filter data in the same packet).

Data polling commands generate two individual reply packets: An ACK/NACK packet and a data packet. Enable/Disable continuous data commands generate an ACK/NACK packet followed by the continuous stream of data packets.

The IMU, GNSS, and Estimation Filter data packets can be set up so that each data quantity is sent at a different rate. For example, you can setup continuous data to send the accelerometer vector at 100Hz and the magnetometer vector at 5Hz. This means that packets will be sent at 100Hz and each one will have the accelerometer vector but only every 20th packet will have the magnetometer vector. This helps reduce bandwidth and buffering requirements. An example of this is given in the IMU Message Format command.

Example Setup Sequence

Setup involves a series of command/reply pairs. The example below demonstrates actual setup sequences that you can send directly to the 3DM-GQ4-45 either programmatically or by using a COM utility. In most cases only minor alterations will be needed to adapt these examples for your application.

Continuous Data Example Command Sequence

Most applications will operate with the 3DM-GQ4-45 sending a continuous data stream. In the following example, the IMU data format is set, followed by the Estimation Filter data format. GNSS data will not be included as we will not be cross-checking against the navigation solution. To reduce the amount of streaming data, if present during the configuration, the device is placed into the idle state while performing the device initialization; when configuration is complete, the required data streams are enabled to bring the device out of idle mode. Finally, the configuration is saved so that it will be loaded on subsequent power-ups, eliminating the need to perform the configuration again.

Step 1: Put the Device in Idle Mode (Disabling the IMU, GNSS, and Estimation Filter data-streams)

Send the "Set To Idle" command to put the device in the idle state (reply is ACK/NACK). This is not required but reduces the parsing burden during initialization and makes visual confirmation of the commands easier:

Step 1	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command Set to Idle	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x02**	N/A	**0xE1**	**0xC7**
Reply ACK/NACK	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x02** Error code: **0x00**	**0xD6**	**0x6C**

Copy-Paste version of the command: "7565 0102 0202 E1C7"

Step 2: Configure the IMU data-stream format

Send a "Set IMU Message Format" command (reply is ACK/NACK). This example requests scaled gyro, scaled accelerometer, and GPS Correlation Timestamp information at 500 Hz (500Hz base rate, with a rate decimation of 1 on the 3DM-GQ4-45 = 500 Hz.) This will result in a single IMU data packet sent at 500 Hz containing the scaled gyro field followed by the scaled accelerometer field followed by the IMU GPS Correlation Timestamp. This is a very typical configuration for a base level of inertial data. If different rates were requested, then each packet would only contain the data quantities that fall in the same decimation frame (see the Multiple Rate Data section). If the stream was not disabled in the previous step, the IMU data would begin stream immediately.

Please note, this command will not append the requested descriptors to the current IMU data-stream configuration, it will overwrite it completely.

Step 2	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command New IMU Message Format	**0x75**	**0x65**	**0x0C**	**0x0D**	**0x0D**	**0x08**	Function: **0x01** Desc count: **0x03** 1st Descriptor: **0x04** Rate Dec: **0x0001** 2nd Descriptor:**0x05** Rate Dec: **0x0001** 3rd Descriptor:**0x12** Rate Dec: **0x0001**	**0x2A**	**0x35**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x08** Error code: **0x00**	**0xE7**	**0xBA**

Copy-Paste version of the command: "7565 0C0D 0D08 0103 0400 0105 00011200 012A 35"

Step 3: Configure the Estimation Filter data-stream format

The following configuration command requests the Estimated LLH Position, Estimated NED Velocity, Estimated Orientation in Quaternion form, and Filter Status at 100 Hz (500Hz base rate, with a rate decimation of 5 = 100 Hz.) This will result in a single Estimation Filter packet sent at 100 Hz containing the requested fields in the requested order. If different rates were requested, the each packet would only contain the data quantities that fall in the same data rate frame (see the Multiple Rate Data section). If the stream was not disabled in the previous step, the Estimation Filter data would begin stream immediately.

Please note, this command will not append the requested descriptors to the current Estimation Filter data-stream configuration, it will overwrite it completely.

Step 3	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command New Estimation Filter Message Format	**0x75**	**0x65**	**0x0C**	**0x10**	**0x10**	**0x0A**	Function: **0x01** Desc Count: **0x04** Est. Pos desc: **0x01** Rate dec: **0x0005** Est.Vel desc: **0x02** Rate dec: **0x0005** Est. Quat desc: **0x03**	**0x3F**	**0x31**

							Rate dec: **0x0005** Filter Status desc: **0x10** Rate dec: **0x0005**		
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x0A** Error code: **0x00**	**0xE9**	**0xBE**

Copy-Paste version of the command: "7565 0C10100A 0104 0100 0502 0005 0300 0510 00053F31"

Step 4: Save the IMU and Estimation Filter MIP Message format

To save the IMU and Estimation Filter MIP Message format, use the "Save" function selector (0x03) in the IMU and Estimation Filter Message Format commands. Below we've combined the two commands as two fields in the same packet. Notice that the two reply ACKs comes in one packet also. Alternatively, they could be sent as separate packets.

Step 4	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command field 1 Save Current IMU Message Format	**0x75**	**0x65**	**0x0C**	**0x08**	**0x04**	**0x08**	Function: **0x03** Desc count: **0x00**		
Command field 2 Save Current Estimation Filter Message Format					**0x04**	**0x0A**	Function: **0x03** Desc count: **0x00**	**0x0E**	**0x31**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x08**	**0x04**	**0xF1**	Cmd echo: **0x08** Error code: **0x00**		
Reply field 2 ACK/NACK					**0x04**	**0xF1**	Cmd echo: **0x0A** Error code: **0x00**	**0xEA**	**0x71**

Copy-Paste version of the command: "7565 0C08 0408 0300 040A 0300 0E31"

Step 5: Enable the IMU and Estimation Filter data-streams

Send an "Enable/Disable Continuous Stream" command to enable the IMU and Estimation Filter continuous streams (reply is ACK). These streams may have already been enabled by default; this step is to confirm they are enabled. These streams will begin streaming data immediately.

Step 5	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command field 1 Enable Continuous IMU Message	**0x75**	**0x65**	**0x0C**	**0x0A**	**0x05**	**0x11**	Fctn: **0x01** IMU: **0x01** ON: **0x01**		
Command field 2 Enable Continuous Estimation Filter Message					**0x05**	**0x11**	Fctn: **0x01** Estimation Filter: **0x03** ON: **0x01**	**0x24**	**0xCC**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x08**	**0x04**	**0xF1**	Cmd echo: **0x11** Error code: **0x00**		
Reply field 2 ACK/NACK					**0x04**	**0xF1**	Cmd echo: **0x11** Error code: **0x00**	**0xFA**	**0xB5**

Copy-Paste version of the command: "7565 0C0A 0511 0101 0105 1101 0301 24 CC"

Step 6 (Optional): Resume the Device

Sending the "Resume" command is another method of re-enabling transmission of enabled data streams (reply is ACK/NACK).

Step 6	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command Resume	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x06**	N/A	**0xE5**	**0xCB**
Reply ACK/NACK	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x06** Error code: **0x00**	**0xDA**	**0x74**

Copy-Paste version of the command: "7565 0102 0206 E5CB"

Step 7: Initialize the Filter

At this point in the set-up, the GQ4-45 is streaming data, but the Kalman Filter is not yet initialized. For a successful initialization to occur the GNSS must have a fix and the initial orientation must be known. The orientation may be initialized in 4 different ways: Setting all of the attitude elements manually, setting only the heading and allowing the device to determine pitch and roll, using the internal IMU solution (which requires the magnetometers) to provide the initial orientation, or via auto-initialization, which uses the chosen heading update source to initialize. In this example, we will assume the magnetometers are available and use the IMU solution to initialize the Kalman Filter. Once the attitude is initialized and the GNSS fix becomes valid, the Kalman Filter estimation will propagate. Note that this step is not necessary if you have the auto-initialize option enabled:

Poll for current Complimentary Filter Euler Angle output:

Step 7a	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command *Poll for CF Euler*	**0x75**	**0x65**	**0x0C**	**0x07**	**0x07**	**0x01**	Fctn: **0x00** Field Count: **0x01** Euler Desc: **0x0C** Reserved: **0x0000**	**0x02**	**0xFC**
Reply *ACK/NACK*	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x01** Error code: **0x00**	**0xE0**	**0xAC**
Reply *Data Packet*	**0x75**	**0x65**	**0x80**	**0x0E**	**0x0E**	**0x0C**	Roll:**0xBAE3ED9B** Pitch:**0x3C7D6DD**F Yaw:**0xBF855CF5**	**0x41**	**0xBB**

Copy-Paste version of the command: "7565 0C07 0701 0001 0C00 0002 FC"

Initialize Attitude:

Step 7b	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command *Initialize Attitude*	**0x75**	**0x65**	**0x0D**	**0x06**	**0x06**	**0x02**	Roll:**0xBAE3ED9B** Pitch:**0x3C7D6DD**F Yaw:**0xBF855CF5**	**0xC4**	**0x09**
Reply *ACK/NACK*	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x02** Error code: **0x00**	**0xE2**	**0xB4**

Copy-Paste version of the command: "7565 0D0E 0E02 BAE3 ED9B 3C7D 6DDF BF85 5CF5 C409"

Polling Data Example Sequence

Polling for data is less efficient than processing a continuous data stream, but may be more appropriate for certain applications. The main difference from the continuous data example is the inclusion of the Poll data commands in the data loop:

Step 1: Put the Device in Idle Mode (Disabling the IMU, GNSS, and Estimation Filter data-streams)

Same as continuous streaming. See above.

Step 2: Configure the IMU data-stream format

Same as continuous streaming. See above.

Step 3: Configure the Estimation Filter data-stream format

Same as continuous streaming. See above.

Step 4: Save the IMU and Estimation Filter MIP Message format

Same as continuous streaming. See above.

Step 5: Resume the Device

Same as continuous streaming step 6. See above.

Step 6: Initialize the Filter

Same as continuous streaming step 7. See above.

Step 7: Send individual data polling commands

Send individual Poll IMU Data and Poll Estimation Filter Data commands in your data collection loop. After the ACK/NACK is sent by the device, a single data packet will be sent according to the settings in the previous steps. Note that the ACK/NACK has the same descriptor set value as the command, but the data packet has the descriptor set value for the type of data (IMU or Estimation Filter):

Step 7	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command Poll IMU Data	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0x01**	Option: **0x00** Desc Count: **0x00**	**0xEF**	**0xDA**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x01** Error code: **0x00**	**0xE0**	**0xAC**
IMU Data Packet field 1 (Gyro Vector)	**0x75**	**0x65**	**0x80**	**0x1C**	**0x0E**	**0x04**	**0x3E 7A 63 A0** **0xBB 8E 3B 29** **0x7F E5 BF 7F**		

IMU Data Packet field 2(Accel Vector)					**0x0E**	**0x03**	**0x3E 7A 63 A0** **0xBB 8E 3B 29** **0x7F E5 BF 7F**	**0xAD**	**0xDC**

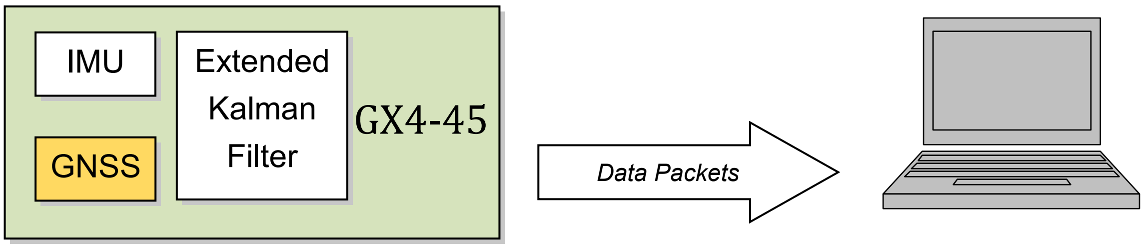
Copy-Paste version of the command: "7565 0C04 0401 0000EF DA"

You may specify the format of the data packet on a per-polling-command basis rather than using the pre-set data format (see the Poll IMU Data and Poll Estimation Filter Data sections*)*

The polling command has an option to suppress the ACK/NACK in order to keep the incoming stream clear of anything except data packets. Set the option byte to 0x01 for this feature.

Parsing Incoming Packets

Setup is usually the easy part of programming the 3DM-GQ4-45. Once you start continuous data streaming, parsing and processing the incoming data packet stream will become the primary focus. The stream of data from the IMU and Kalman Filter (Estimation Filter) are usually the dominant source of data since they come in the fastest. Polling for data may seem to be a logical solution to controlling the data flow, and this may be appropriate for some applications, but if your application requires the precise delivery of inertial data, it is often necessary to have the data stream drive the process rather than having the host try to control the data stream through polling.



The "descriptor set" qualifier in the MIP packet header is a feature that greatly aids the management of the incoming packet stream by making it easy to sort the packets into logical sub-streams and route those streams to appropriate handlers. The first step is to parse the incoming character stream into packets.

It is important to take an organized approach to parsing continuous data. The basic strategy is this: parse the incoming stream of characters for the packet starting sequence "ue" and then wait for the entire packet to come in based on the packet length byte which arrives after the "ue" and descriptor set byte. Make sure you have a timeout on your wait loop in case your stream is out of sync and the starting "ue" sequence winds up being a "ghost" sequence. If you timeout, restart the parsing with the first character after the ghost "ue". Once the stream is in sync, it is rare that you will hit a timeout unless you have an unreliable communications link. After verifying the checksum, examine the "descriptor set" field in the header of the packet. This tells you immediately how to handle the packet.

Based on the value of the descriptor set field in the packet header, pass the packet to either a command handler (if it is a Base command or 3DM command descriptor set) or a data handler (if it is a GNSS, IMU, or Estimation Filter data set). Since you know beforehand that the IMU and Estimation Filter data packets will be coming in fastest, you can tune your code to buffer or handle these packets at a high priority. Likewise, you know that the GNSS packets will be coming in at a much lower rate but may have much more data to process. Again, you can tune your code to buffer or handle these slower packets appropriately. Replies to commands generally happen sequentially after a command so the incidence of these is under program control.

For multithreaded applications, it is often useful to use queues to buffer packets bound for different packet handler threads. The depth of the queue can be tuned so that no packets are dropped while waiting for their associated threads to process the packets in the queue. See Advanced Programming Models section for more information on this topic.

Once you have sorted the different packets and sent them to the proper packet handler, the packet handler may parse the packet payload fields and handle each of the fields as appropriate for the application. For simple applications, it is perfectly acceptable to have a single handler for all packet types. Likewise, it is perfectly acceptable for a single parser to handle both the packet type and the fields in the packet. The ability to sort the packets by type is just an option that simplifies the implementation of more sophisticated applications.

Multiple Rate Data

The message format commands (IMU Message Format, GNSS Message Format, and Estimation Filter Message Format) allow you to set different data rates for different data quantities. This is a very useful feature especially for IMU data because some data, such as accelerometer and gyroscope data, usually requires higher data rates (>100Hz) than other IMU data such as Magnetometer (20Hz typical) data. The ability to send data at different rates reduces the parsing load on the user program and decreases the bandwidth requirements of the communications channel.

Multiple rate data is scheduled on a common sampling rate clock. This means that if there is more than one data rate scheduled, the schedules coincide periodically. For example, if you request Accelerometer data at 100Hz and Magnetometer data at 50Hz, the magnetometer schedule coincides with the Accelerometer schedule 50% of the time. When the schedules coincide, then the two data quantities are delivered in the same packet. In other words, in this example, you will receive data packets at 100Hz and every packet will have an accelerometer data field and EVERY OTHER packet will also include a magnetometer data field:

Packet 1	*Packet 2*	*Packet 3*	*Packet 4*	*Packet 5*	*Packet 6*	*Packet 7*	*Packet 8*	*….*
Accel	Accel Mag	Accel	Accel Mag	Accel	Accel Mag	Accel	Accel Mag	Accel

If a timestamp is included at 100Hz, then the timestamp will also be included in every packet in this example. It is important to note that *the data in a packet with a timestamp is always synchronous with the timestamp.* This assures that multiple rate data is always synchronous.

Packet 1	*Packet 2*	*Packet 3*	*Packet 4*	*Packet 5*	*Packet 6*	*….*
Accel Timestamp	Accel Mag Timestamp	Accel Timestamp	Accel Mag Timestamp	Accel Timestamp	Accel Mag Timestamp	Accel Timestamp

Data Synchronicity

Because the MIP packet allows multiple data fields to be in a single packet, it may be assumed that a single timestamp field in the packet applies to all the data in the packet. In other words, it may be assumed that all the data fields in the packet were sampled at the same time.

IMU, GNSS, and Estimation Filter data are generated independently by three systems with different clocks. The importance of time is different in each system and the data they produce. The IMU data requires precise microsecond resolution and perfectly regular intervals in its timestamps. GNSS data produces very precise UTC interval data but it is typically delivered in a 1 second time frame. The Kalman Filter resides on a separate processor and must derive its timing information from the two data sources.

The time base difference is one of the factors that necessitate separation of the GNSS, IMU, and Estimation Filter data into separate packets. Conversely, the common time base of the different data quantities within one system is what allows grouping multiple data quantities into a single packet with a common timestamp. In other words, IMU data is always grouped with a timestamp generated from the IMU time base, and GNSS data is always grouped with a timestamp from the GPS time base, etc.

All data streams (IMU, GNSS, and Estimation Filter) on the 3DM-GQ4-45 output a “GPS Time”-formatted timestamp. This timestamp is synchronized between the 3 devices using the GNSS receiver 1PPS (one pulse per second) hardware beacon. This allows a precise common time base for all data. Due to the differences in clocks on each device, the period between two consecutive timestamp values may not be constant; this occurs because periodic corrections are applied to the IMU and Estimation Filter timestamps when the GNSS receiver 1PPS signal is asserted.

Due to the introduction of new satellite constellations, the collective moniker “GNSS” has been adopted as a blanket term to encompass all navigation satellite constellations. However, “GPS Time” refers to the specific time base used by U.S. GPS satellites. GLONASS and BeiDou and other constellations do not use this exact time format, however the GNSS receiver used in the 3DM-GQ4 converts the time bases to synchronize with the GPS time base. In this manual we refer to the receiver as a “GNSS” receiver and the timestamps as “GPS Time” timestamps.

Communications Bandwidth Management

Because of the large amount and variety of data that is available from the 3DM-GQ4-45, it is quite easy to overdrive the bandwidth of the communications channel. This can result in dropped packets. The 3DM-GQ4-45 does not do analysis of the bandwidth requirements for any given output data configuration, it will simply drop a packet if its internal serial buffer is being filled faster than it is being emptied. It is up to the programmer to analyze the size of the data packets requested and the available bandwidth of the communications channel. Often the best way to determine this is empirically by trying different settings and watching for dropped packets. Below are some guidelines on how to determine maximum bandwidth for your application.

UART Bandwidth Calculation

Below is an equation for the maximum theoretical UART BAUD rate for a given message configuration. Although it is possible to calculate the approximate bandwidth required for a given setup, there is no guarantee that the system can support that setup due to internal processing delays. The best approach is to try a setting based on an initial estimate and watch for dropped packets. If there are dropped packets, increase the BAUD rate, reduce the data rate, or decrease the size or number of packets.

$$\mathrm{n}(\mathrm{k} \times f_{mr}) + \mathrm{n}\sum(S_f \times f_{dr})$$

Where

$$S_f = Size\ of\ data\ field\ in\ bytes$$
$$f_{dr} = field\ data\ rate\ in\ Hz$$
$$f_{mr} = maximum\ data\ rate\ in\ Hz$$
$$\mathrm{n} = size\ of\ UART\ word = 10 bits$$
$$\mathrm{k} = Size\ of\ MIP\ wrapper = 6\ bytes$$

which becomes

$$60 f_{mr} + 10\sum(S_f \times f_{dr})$$

Example:

For an IMU message format of Accelerometer Vector (14 byte data field) + Internal Timestamp (6 byte data field), both at 100 Hz, the theoretical minimum BAUD rate would be:

$$= 60 \times 100 + 10((14 \times 100) + (6 \times 100))$$

$$= 26000\ \mathrm{BAUD}$$

In practice, if you set the BAUD rate to 115200 the packets come through without any packet drops. If you set the BAUD rate to the next available lower rate of 19200, which is lower than the calculated minimum, you get regular packet drops. The only way to determine a packet drop is by observing a timestamp in sequential packets. The interval should not change from packet to packet. If it does change then packets were dropped.

USB vs. UART

The 3DM-GQ4-45 has a dual communication interface: USB or UART. There is an important difference between USB and UART communication with regards to data bandwidth. The USB "virtual COM port" that the 3DM-GQ4-45 implements runs at USB "full-speed" setting of 12Mbs (megabits per second). However, USB is a polled master-slave system and so the slave (3DM-GQ4-45) can only communicate when polled by the master. This results in inconsistent data streaming – that is, the data comes in spurts rather than at a constant rate and, although rare, sometimes data can be dropped if the host processor fails to poll the USB device in a timely manner.

With the UART the opposite is true. The 3DM-GQ4-45 operates without UART handshaking which means it streams data out at a very consistent rate without stopping. Since the host processor has no handshake method of pausing the stream, it must instead make sure that it can process the incoming packet stream non-stop without dropping packets.

In practice, USB and UART communications behave similarly on a Windows based PC, however, UART is the preferred communications system if consistent, deterministic communications timing behavior is required. USB is preferred if you require more data than is possible over the UART and you can tolerate the possibility of variable latency in the data delivery and very occasional packet drops due to host system delays in servicing the USB port.

Command Reference

Base Commands

The Base command set is common to many MicroStrain devices. With the Base command set it is possible to identify many properties and do basic functions on a device even if you do not recognize its specialized functionality or data. The commands work the same way on all devices that implement this set.

Ping (0x01, 0x01)

Description	Send a “Ping” command		
Notes	Device responds with ACK/NACK packet if present.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x01	N/A
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, non-zero:NACK)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Ping	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x01**		**0xE0**	**0xC6**
Reply ACK/NACK	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Command echo: **0x01** Error code: **0x00**	**0xD5**	**0x6A**

Copy-Paste version of the command: “7565 0102 0201 E0C6”

Set To Idle (0x01, 0x02)

Description	Place device into idle mode.		
Notes	Command has no parameters. Device responds with ACK if successfully placed in idle mode. This command will suspend streaming (if enabled) or wake the device from sleep (if sleeping) to allow it to respond to status and setup commands. You may restore the device mode by issuing the Resume command.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x02	N/A
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, non-zero:NACK)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Set To Idle	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x02**		**0xE1**	**0xC7**
Reply ACK/NACK	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Command echo: **0x02** Error code: **0x00**	**0xD6**	**0x6C**

Copy-Paste version of the command: "7565 0102 0202 E1C7"

Resume (0x01, 0x06)

Description	Place device back into the mode it was in before issuing the Set To Idle command. If the Set To Idle command was not issued, then the device is placed in default mode.		
Notes	Command has no parameters. Device responds with ACK if stream successfully enabled.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x06	N/A
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0: ACK, non-zero: NACK)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Set To Idle	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x06**		**0xE5**	**0xCB**
Reply ACK/NACK	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Command echo: **0x06** Error code: **0x00**	**0xDA**	**0x74**

Copy-Paste version of the command: "7565 0102 0206 E5CB"

GPS Time Update (0x01, 0x72)

Description	This message updates the internal GPS Time as reported in the Filter Timestamp.		
Notes	This command enables synchronization of IMU/AHRS Timestamps with an external GNSS receiver. When combined with a PPS input applied to pin 7 of the i/o connector, the GPS Correlation Timestamp in the inertial data output is synchronized with the external GNSS clock. It is recommended that this update command be sent once per second. See the GPS Correlation Timestamp for more information. *Possible function selector values:* 0x01 – Apply new settings 0x02 – Read back current settings. 0x06 – Apply new settings with no ACK/NACK Reply *Possible field selector values:* 0x01 – GPS Week Number. 0x02 – GNSS Seconds.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x08	0x72	U8 – Function Selector U8 – GPS Time Field Selector U32 – New Time Value
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0: ACK, non-zero: NACK)
Reply field 2 (function = 2 selector = 1)	0x06	0x84	U32 – Current GPS Week Value
Reply field 2 (function = 2 selector = 2)	0x06	0x85	U32 – Current GPS Seconds Value
Example	MIP Packet Header	Command/Reply Fields	Checksum

	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command GPS Time Update	**0x75**	**0x65**	**0x01**	**0x08**	**0x08**	**0x72**	*Fctn(Apply)*:**0x01** *Field (Week):* **0x01** *Val*:**0x00000698**	**0xFD**	**0x32**
Reply ACK/NACK	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Cmd echo: **0x72** Error code: **0x00**	**0x46**	**0x4C**

Copy-Paste version of the command: “7565 0108 0872 0101 0000 0698 FD32”

Get Device Information (0x01, 0x03)

Description	Get the device ID strings and firmware version					
Notes	Reply has two fields: “ACK/NACK” and “Device Info Field”					
Field Format	*Field Length*	*Field Descriptor*	*Field Data*			
Command	0x02	0x03	N/A			
Replyfield 1 ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0: ACK, non-zero: NACK)			
Reply field 2 Device Info Field	0x52	0x81	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	Firmware Version	U16	N/A
			2	Model Name	String(16)	N/A
			18	Model Number	String(16)	N/A
			34	Serial Number	String(16)	N/A
			50	Lot Number	String(16)	N/A
			66	Device Options	String(16)	N/A

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Get Device Info	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x03**		**0xE2**	**0xC8**
ReplyField 1 ACK/NACK	**0x75**	**0x65**	**0x01**	**0x58**	**0x04**	**0xF1**	Command echo: **0x03** Error code: **0x00**		
Reply Field 2 Device Info Field					**0x54**	**0x81**	FW Version: **0x05FE** "3DM-GQ4-45" "6250-4220" "6250-23465" " " "5g, 300dps"	**0x##**	**0x##**

Copy-Paste version of the command: “7565 0102 0203 E2C8”

Get Device Descriptor Sets (0x01, 0x04)

Description	Get the set of descriptors that this device supports		
Notes	Reply has two fields: "ACK/NACK" and "Descriptors". The "Descriptors" field is an arrayof 16 bit values. The MSB specifies the descriptor set and the LSB specifies the descriptor.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x04	N/A
Replyfield 1 *ACK/NACK*	0x04	0xF1	U8 – echo the command byte U8 – error code (0: ACK, non-zero: NACK)
Reply field 2 *Array of Descriptors*	2 x <Number of descriptors> + 2	0x82	*Binary Offset* / *Description* / *Data Type* 0 / MSB: Descriptor Set, LSB: Descriptor / U16 1 / MSB: Descriptor Set, LSB: Descriptor / U16 … / *<etc>* / …

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command *Get Device Info*	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x04**		**0xE3**	**0xC9**
ReplyField 1 *ACK/NACK*	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Command echo: **0x04** Error code: **0x00**		
Reply Field 2 *Array of Descriptors*					***<n*2>***	**0x82**	**0x0101** **0x0102** **0x0103** **…** **0x0C01** **0x0C02** **…** *nth descriptor:* **0x0C72**	**0x##**	**0x##**

Copy-Paste version of the command: "7565 0102 0204 E3C9"

Device Built-In Test (0x01, 0x05)

Description	Run the device Built-In Test (BIT). The Built-In Test command always returns a 32 bit value. A value of 0 means that all tests passed. A non-zero value indicates that not all tests passed. The failure flags are device dependent. The flags for the 3DM-GQ4-45 are defined below.		
Notes	**3DM-GQ4-45 BIT Error Flags:** (see table below)		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x05	N/A
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, non-zero: NACK)
Reply field 2 BIT Error Flags	0x06	0x83	U32 – BIT Error Flags

3DM-GQ4-45 BIT Error Flags:

Byte	Byte 1 (LSB)	Byte 2	Byte 3	Byte 4 (MSB)
Device	Processor Board	Sensor Board	GNSS	Kalman Filter
Bit 1 (LSB)	WDT Reset (Latching, Reset after first commanded BIT)	IMU Communication Fault	GNSS Power Fault	Solution Fault
Bit 2	Input voltage fault	Reserved	GNSS Communication Fault	Reserved
Bit 3	System voltage fault	Reserved	GNSS Solution Fault	Reserved
Bit 4	Temp sensor fault	Reserved	Antenna short fault	Reserved
Bit 5	Reserved	Magnetometer Fault (if applicable)	Reserved	Reserved
Bit 6	Reserved	Pressure Sensor Fault (if applicable)	Reserved	Reserved
Bit 7	Reserved	Reserved	Reserved	Reserved
Bit 8 (MSB)	Reserved	Reserved	Reserved	Reserved

Example

MIP Packet Header				Command/Reply Fields			Checksum	
Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*

LORD MicroStrain® SENSING SYSTEMS

Command Built-In Test	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x05**	N/A	**0xE4**	**0xCA**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x01**	**0x0A**	**0x04**	**0xF1**	*Echo cmd:* **0x05** *Error code:* **0x00**		
Reply field 2 BIT Error Flags					**0x06**	**0x83**	*BIT Error Flags:* **0x00000000**	**0x68**	**0x7D**

Copy-Paste version of the command: "7565 0102 0205 E4CA"

Device Reset (0x01, 0x7E)

Description	Resets the Device.		
Notes	Device responds with ACK if it recognizes the command and then immediately resets.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x7E	N/A
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0: ACK, non-zero: NACK)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Set Reset	**0x75**	**0x65**	**0x01**	**0x02**	**0x02**	**0x7E**	*N/A*	**0x5D**	**0x43**
Reply ACK/NACK	**0x75**	**0x65**	**0x01**	**0x04**	**0x04**	**0xF1**	Command echo: **0x7E** Error code: **0x00**	**0x52**	**0x64**

Copy-Paste version of the command: "7565 0102 027E 5D43"

3DM Commands

The 3DM command set is common to the MicroStrain Inertial sensors that support the MIP packet protocol. Because of the unified set of commands, it is easy to migrate code from one inertial sensor to another.

Poll IMU Data (0x0C, 0x01)

Description	Poll the device for an IMU message with the specified format		
Notes	This function polls for an IMU message using the provided format. The resulting message will maintain the order of descriptors sent in the command and any unrecognized descriptors are ignored. If the format is not provided, the device will attempt to use the stored format (set with the Set IMU Message Format command.) If no format is provided and there is no stored format, the device will respond with a NACK. The reply packet contains an ACK/NACK field. The polled data packet is sent separately as an IMU Data packet. *Possible Option Selector Values:* 0x00 – Normal ACK/NACK Reply. 0x01 – Suppress the ACK/NACK reply.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	4 + 3*N	0x01	U8 – Option Selector U8 – Number of Descriptors (N), N*(U8 – Descriptor, U16 Reserved)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Poll IMU data (use stored format)	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0x01**	Option: **0x00** Desc count: **0x00**	**0xEF**	**0xDA**
Command Poll IMU data (use specified format)	**0x75**	**0x65**	**0x0C**	**0x0A**	**0x0A**	**0x01**	Option: **0x00** Desc count: **0x02** 1st Descriptor: **0x04** Reserved: **0x0000** 2nd Descriptor:**0x05** Reserved: **0x0000**	**0x06**	**0x27**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x01** *Error code:***0x00**	**0xE0**	**0xAC**

Copy-Paste versions of the commands:
Stored format: “7565 0C04 0401 0000 EFDA”
Specified format: “7565 0C0A 0A01 0002 0400 0005 0000 0627”

Poll GNSS Data (0x0C, 0x02)

<table>
<tr><td>Description</td><td colspan="9">Poll the device for a GNSS message with the specified format</td></tr>
<tr><td>Notes</td><td colspan="9">This function polls for a GNSS message using the provided format. The resulting message will maintain the order of descriptors sent in the command and any unrecognized descriptors are ignored. If the format is not provided, the device will attempt to use the stored format (set with the Set GNSS Message Format command.) If no format is provided and there is no stored format, the device will respond with a NACK. The reply packet contains an ACK/NACK field. The polled data packet is sent separately as a GNSS Data packet.

Possible Option Selector Values:
0x00 – Normal ACK/NACK Reply.
0x01 – Suppress the ACK/NACK reply.</td></tr>
<tr><td>Field Format</td><td colspan="2">Field Length</td><td colspan="3">Field Descriptor</td><td colspan="4">Field Data</td></tr>
<tr><td>Command</td><td colspan="2">4 + 3*N</td><td colspan="3">0x02</td><td colspan="4">U8 – Option Selector
U8 – Number of Descriptors (N),
N*(U8 – Descriptor, U16 Reserved)</td></tr>
<tr><td>Reply ACK/NACK</td><td colspan="2">0x04</td><td colspan="3">0xF1</td><td colspan="4">U8 – echo the command byte
U8 – error code (0:ACK, not 0:NACK)</td></tr>
<tr><td rowspan="2">Examples</td><td colspan="4">MIP Packet Header</td><td colspan="3">Command/Reply Fields</td><td colspan="2">Checksum</td></tr>
<tr><td>Sync1</td><td>Sync2</td><td>Desc Set</td><td>Payload Length</td><td>Field Length</td><td>Field Desc.</td><td>Field Data</td><td>MSB</td><td>LSB</td></tr>
<tr><td>Command
Poll GNSS data (use stored format)</td><td>0x75</td><td>0x65</td><td>0x0C</td><td>0x04</td><td>0x04</td><td>0x02</td><td>Option: 0x00
Desc count: 0x00</td><td>0xF0</td><td>0xDD</td></tr>
<tr><td>Command
Poll GNSS data (use specified format)</td><td>0x75</td><td>0x65</td><td>0x0C</td><td>0x0A</td><td>0x0A</td><td>0x02</td><td>Option: 0x00
Desc count: 0x02
1st Descriptor: 0x03
Reserved: 0x0000
2nd Descriptor:0x05
Reserved: 0x0000</td><td>0x06</td><td>0x2A</td></tr>
<tr><td>Reply
ACK/NACK (Data</td><td>0x75</td><td>0x65</td><td>0x0C</td><td>0x04</td><td>0x04</td><td>0xF1</td><td>Echo cmd:0x02
Error code:0x00</td><td>0xE1</td><td>0xAE</td></tr>
</table>

packet is sent separately if ACK)									

Copy-Paste versions of the commands:
Stored format: "7565 0C04 0402 0000 F0DD"
Specified format: "7565 0C0A 0A02 0002 0300 0005 0000 062A"

Poll Estimation Filter Data (0x0C, 0x03)

Description	Poll the device for a Estimation Filter message with the specified format		
Notes	This function polls for a Estimation Filter message using the provided format. The resulting message will maintain the order of descriptors sent in the command and any unrecognized descriptors are ignored. If the format is not provided, the device will attempt to use the stored format (set with the Set Estimation Filter Message Format command.) If no format is provided and there is no stored format, the device will respond with a NACK. The reply packet contains an ACK/NACK field. The polled data packet is sent separately as a Estimation Filter Data packet. *Possible Option Selector Values:* 0x00 – Normal ACK/NACK Reply. 0x01 – Suppress the ACK/NACK reply.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	4 + 3*N	0x03	U8 – Option Selector U8 – Number of Descriptors (N), N*(U8 – Descriptor, U16 Reserved)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Poll Estimation Filter data (use stored format)	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0x03**	Option: **0x00** Desc count: **0x00**	**0xF1**	**0xE0**
Command Poll Estimation	**0x75**	**0x65**	**0x0C**	**0x0A**	**0x0A**	**0x03**	Option: **0x00** Desc count: **0x02**	**0x02**	**0x1E**

Filter data (use specified format)							1st Descriptor: **0x01** Reserved: **0x0000** 2nd Descriptor:**0x02** Reserved: **0x0000**		
Reply ACK/NACK (Data packet is sent separately if ACK)	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x03** *Error code:***0x00**	**0xE2**	**0xB0**

Copy-Paste versions of the commands:
Stored format: "7565 0C04 0403 0000 F1E0"
Specified format: "7565 0C0A 0A03 0002 0100 0002 0000 021E"

Get IMU Data Base Rate (0x0C, 0x06)

Description	Get the base rate for the IMU data in Hz.		
Notes	Returns the value used for data rate calculations. See the IMU Message Format command.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x06	*none*
Reply field 1 ACK/NACK Field	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)
Reply field 2 IMU Base Rate	0x04	0x83	U16 – IMU data base rate (Hz)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	MSB	LSB
Command Get IMU Base Rate	**0x75**	**0x65**	**0x0C**	**0x02**	**0x02**	**0x06**		**0xF0**	**0xF7**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x08**	**0x04**	**0xF1**	*Echo cmd:* **0x06** *Error code:* **0x00**		
Reply field 2 IMU Base Rate					**0x04**	**0x83**	Base rate (Hz): **0x0064**	**0xD4**	**0x6B**

Copy-Paste version of the command: "7565 0C02 0206 F0F7"

Get GNSS Data Base Rate (0x0C, 0x07)

Description	Get the base rate for the GNSS data in Hz.		
Notes	Returns the value used for data rate calculations. See the GNSS Message Format command.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x06	*none*
Reply field 1 *ACK/NACK Field*	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)
Reply field 2 *GNSS Base Rate*	0x04	0x84	U16 – GNSS data base rate (Hz)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	MSB	LSB
Command *Get GNSS Base Rate*	**0x75**	**0x65**	**0x0C**	**0x02**	**0x02**	**0x07**		**0xF1**	**0xF8**
Reply field 1 *ACK/NACK*	**0x75**	**0x65**	**0x0C**	**0x08**	**0x04**	**0xF1**	*Echo cmd:* **0x07** *Error code:* **0x00**		
Reply field 2 *GNSS Base Rate*					**0x04**	**0x84**	Base rate (Hz): **0x0004**	**0x76**	**0x14**

Copy-Paste version of the command: "7565 0C02 0207 F1F8"

Get Estimation Filter Data Base Rate (0x0C, 0x0B)

Description	Get the base rate for the Estimation Filter data in Hz.		
Notes	Returns the value used for data rate calculations. See the Estimation Filter Message Format command.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x0B	*none*
Reply field 1 ACK/NACK Field	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)
Reply field 2 Estimation Filter Base Rate	0x04	0x8A	U16 – GNSS data base rate (Hz)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	MSB	LSB
Command Get GNSS Base Rate	**0x75**	**0x65**	**0x0C**	**0x02**	**0x02**	**0x0B**		**0xF5**	**0xFC**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x08**	**0x04**	**0xF1**	*Echo cmd:* **0x0B** *Error code:* **0x00**		
Reply field 2 Estimation Filter Base Rate					**0x04**	**0x8A**	Base rate (Hz): **0x0064**	**0xE0**	**0x9E**

Copy-Paste version of the command: "7565 0C02 020B F5FC"

IMU Message Format (0x0C, 0x08)

Description	Set, read, or save the format of the IMU message packet. This command sets the format for the IMU data packet when in standard mode. The resulting data messages will maintain the order of descriptors sent in the command. The command has a function selector and a descriptor array as parameters.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings The rate decimation field is calculated as follows for IMU messages: Rate Decimation = IMU Base Rate / Desired Data Rate You should always retrieve the Base Rate from the Get IMU Data Base Rate command for computing the desired rate decimation. Base rates vary from device to device. The IMU base rate for the 3DM-GQ4 is 500. The device checks that all descriptors are valid prior to executing this command. If any of the descriptors are invalid for the IMU descriptor set, a NACK will be returned and the message format will be unchanged. The descriptor array only needs to be provided if the function selector is = 1 (Use new settings). For all other functions it may be empty (Number of Descriptors = 0).		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	4 + 3*N	0x08	U8 - Function Selector U8 – Number of Descriptors (N), N*(U8 – Descriptor, U16 – Rate Decimation)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	3 + 3*N	0x80	U8 – Number of Descriptors (N), N*(U8 – Descriptor, U16 – Rate Decimation)

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command IMU Message Format (use new settings)	**0x75**	**0x65**	**0x0C**	**0x0A**	**0x0A**	**0x08**	Function: **0x01** Desc count: **0x02** 1st Descriptor: **0x04** Rate Dec: **0x000A** 2nd Descriptor:**0x05** Rate Dec: **0x000A**	**0x22**	**0xA0**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x08** *Error code:***0x00**	**0xE7**	**0xBA**
Command IMU Message Format (read back current settings)	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0x08**	Function: **0x02** Desc count: **0x00**	**0xF8**	**0xF3**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x0D**	**0x04**	**0xF1**	*Echo cmd:***0x08** *Error code:***0x00**		
Reply field 2 Current IMU Message Format					**0x09**	**0x80**	Desc count: **0x02** 1st Descriptor: **0x03** Rate Dec: **0x000A** 2nd Descriptor:**0x04** Rate Dec: **0x000A**	**0x98**	**0x0F**

Copy-Paste version of the commands:
Use New Settings:"7565 0C0A 0A08 0102 0400 0A05 000A 22A0"
Read Current Settings: "7565 0C04 0408 0200 F8F3"

GNSS Message Format (0x0C, 0x09)

Description	Set, read, or save the format of the GNSS message packet. This function sets the format for the GNSS MIP data packet when in standard mode. The resulting message will maintain the order of descriptors sent in the command. The command has a function selector and a descriptor array as parameters.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings The rate decimation field is calculated as follows for GNSS messages: Rate Decimation = GNSS Base Rate / Desired Data Rate You should always retrieve the Base Rate from the Get GNSS Data Base Rate command for computing the desired rate decimation. Base rates vary from device to device. The GNSS base rate for the 3DM-GQ4 is 4. The device checks that all descriptors are valid prior to executing this command. If any of the descriptors are invalid for the GNSS data descriptor set, a NACK will be returned and the message format will be unchanged.The descriptor array only needs to be provided if the function selector is = 1 (Use new settings). For all other functions it may be empty (Number of Descriptors = 0).		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	4 + 3*N	0x09	U8 - Function Selector U8 – Number of Descriptors (N), N*(U8 – Descriptor, U16 – Rate Decimation)
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2	3 + 3*N	0x81	U8 – Number of Descriptors (N),

(function = 2)			N*(U8 – Descriptor, U16 – Rate Decimation)

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command GNSS Message Format (use new settings)	**0x75**	**0x65**	**0x0C**	**0x0A**	**0x0A**	**0x09**	Function: **0x01** Desc count: **0x02** 1st Descriptor: **0x03** Data rate: **0x0004** 2nd Descriptor:**0x05** Data rate: **0x0004**	**0x16**	**0x85**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x09** *Error code:***0x00**	**0xE8**	**0xBC**
Command GNSS Message Format (read back current settings)	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0x09**	Function: **0x02** Desc count: **0x00**	**0xF9**	**0xF6**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x0D**	**0x04**	**0xF1**	*Echo cmd:***0x09** *Error code:***0x00**		
Reply field 2 Current GNSS Message Format					**0x09**	**0x81**	Desc count: **0x02** 1st Descriptor: **0x03** Data rate: **0x0004** 2nd Descriptor:**0x05** Datarate: **0x0004**	**0x8D**	**0xFE**

Copy-Paste version of the commands:
Use New Settings: "7565 0C0A 0A09 0102 0300 0405 0004 1685"
Read Current Settings: "7565 0C04 0409 0200 F9F6"

Estimation Filter Message Format (0x0C, 0x0A)

<table>
<tr><td>Description</td><td colspan="3">Set, read, or save the format of the Estimation Filter message packet. This function sets the format for the Estimation Filter MIP data packet when in standard mode. The resulting message will maintain the order of descriptors sent in the command. The command has a function selector and a descriptor array as parameters.</td></tr>
<tr><td>Notes</td><td colspan="3">Possible function selector values:

0x01 – Use new settings
0x02 – Read back current settings.
0x03 – Save current settings as startup settings
0x04 – Load saved startup settings
0x05 – Reset to factory default settings

The rate decimation field is calculated as follows for Estimation Filter messages:

Rate Decimation = EF Base Rate / Desired Data Rate

You should always retrieve the Base Rate from the Get Estimation Filter Data Base Rate command for computing the desired rate decimation. Base rates vary from device to device. The EF base rate for the 3DM-GQ4 is 500.

The device checks that all descriptors are valid prior to executing this command. If any of the descriptors are invalid for the Estimation Filter data descriptor set, a NACK will be returned and the message format will be unchanged.The descriptor array only needs to be provided if the function selector is = 1 (Use new settings). For all other functions it may be empty (Number of Descriptors = 0).</td></tr>
<tr><td>Field Format</td><td>Field Length</td><td>Field Descriptor</td><td>Field Data</td></tr>
<tr><td>Command</td><td>4 + 3*N</td><td>0x0A</td><td>U8 - Function Selector
U8 – Number of Descriptors (N),
N*(U8 – Descriptor, U16 – Rate Decimation)</td></tr>
<tr><td>Reply field 1 ACK/NACK</td><td>0x04</td><td>0xF1</td><td>U8 – echo the command descriptor
U8 – error code (0:ACK, not 0:NACK)</td></tr>
</table>

Reply field 2 (function = 2)	3 + 3*N	0x82	U8 – Number of Descriptors (N), N*(U8 – Descriptor, U16 – Rate Decimation)

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Estimation Filter Message Format (use new settings)	**0x75**	**0x65**	**0x0C**	**0x0A**	**0x0A**	**0x0A**	Function: **0x01** Desc count: **0x02** 1st Descriptor: **0x01** Data rate: **0x0001** 2nd Descriptor:**0x02** Data rate: **0x0001**	**0x0C**	**0x6A**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x0A** *Error code:***0x00**	**0xE9**	**0xBE**
Command Estimation Filter Message Format (read back current settings)	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0x0A**	Function: **0x02** Desc count: **0x00**	**0xFA**	**0xF9**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x0D**	**0x04**	**0xF1**	*Echo cmd:***0x0A** *Error code:***0x00**		
Reply field 2 Current GNSS Message Format					**0x09**	**0x82**	Desc count: **0x02** 1st Descriptor: **0x01** Data rate: **0x0001** 2nd Descriptor:**0x02** Datarate: **0x0001**	**0x84**	**0xED**

Copy-Paste version of the commands:
Use New Settings: "7565 0C0A 0A0A 0102 0100 0102 0C6A"
Read Current Settings: "7565 0C04 040A 0200 FAF9"

Enable/Disable Continuous Data Stream (0x0C, 0x11)

Description	Control the streaming of IMU, GNSS, and Estimation Filter data. If disabled, the data from the selected device is not continuously transmitted. Upon enabling, the most current data will be transmitted (i.e. no stale data is transmitted.) The default for the device is all streams enabled. For all functions except 0x01 (use new setting), the new enable flag value is ignored.		
Notes	Possible function selector values: 0x01 – Apply new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Load factory default settings *The device selector can be:* 0x01 – IMU 0x02 – GNSS 0x03 – Estimation Filter *The enable flag can be either:* 0x00 – disable the selected stream. 0x01 – enable the selected stream. *(default)*		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x05	0x11	U8 – Function Selector U8 – Device Selector U8 – New Enable Flag
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x04	0x85	U8 – Device Selector U8 – Current Device Enable Flag

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command IMU Stream ON	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0x11**	Function(Apply):**0x01** Device (IMU): **0x01**	**0x04**	**0x1A**

							Stream (ON): **0x01**		
Command IMU Stream OFF	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0x11**	Function(Apply):**0x01** Device (IMU): **0x01** Stream (OFF): **0x00**	**0x03**	**0x19**
Command GNSS Stream ON	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0x11**	Function(Apply):**0x01** Device (GNSS): **0x02** Stream (ON): **0x01**	**0x05**	**0x1C**
Command GNSS Stream OFF	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0x11**	Function(Apply):**0x01** Device (GNSS): **0x02** Stream (OFF): **0x00**	**0x04**	**0x1B**
Command Estimation Filter Stream ON	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0x11**	Function(Apply):**0x01** Device (Estimation Filter): **0x03** Stream (ON): **0x01**	**0x06**	**0x1E**
Command Estimation Filter Stream OFF	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0x11**	Function(Apply):**0x01** Device (Estimation Filter): **0x03** Stream (OFF): **0x00**	**0x05**	**0x1D**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0xF1**	*Echo cmd:***0x11** *Error code:***0x00**	**0xEF**	**0xCA**

Copy-Paste version of the 1st command: "7565 0C05 0511 0101 0104 1A"

GNSS Constellation Settings (0x0C, 0x21)

Description	This configures which satellite constellations and how many satellites in each constellation the receiver should track.
Notes	*Function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Maximum number of tracking channels to use (total for all constellations):* 0 -> [Number of available channels] (from reply message) *For each constellation you wish to configure include the following sub-fields:* *Constellation ID:* 0 – GPS (G1-G32) 1 – SBAS (S120-S158) 3 – BeiDou (B1-B37) 5 – QZSS (Q1-Q5) 6 – GLONASS (R1-R32) *Constellation Enable:* 0x00 – Disable 0x01 – Enable *Number of tracking channels* *(number of reserved channels for all constellations must total <= 32):* 0 -> 32 Number of reserved channels 0 -> 32 Max number of channels (>= reserved channels) *Constellation Options:* 0x0001 – L1SAIF *(QZSS only)*

Factory default setting is GPS and GLONASS enabled with min/max of GPS 8/16, GLONASS 8/14, SBAS 1/3, QZSS 0/3.

Any setting that causes the total reserved channels to exceed 32 will result in a NACK.

You cannot enable GLONASS and BeiDou at the same time.

Enabling SBAS and QZSS only augments GPS accuracy.

It is recommended to disable GLONASS and BeiDou if a GPS-only antenna or GPS-only SAW filter is used.

This u-blox SBAS implementation is, in accordance with standard RTCA/DO-229D, a class Beta-1 equipment. All timeouts etc. are chosen for the En-Route Case. Do not use this equipment under any circumstances for "safety of life" applications!

	Field Length	*Field Descriptor*	*Field Data*
Command	6 + 6*N	0x21	U8 – Function selector U16 – Maximum channels to use U8 – Count of constellations to configure (N) N * (U8 – Constellation ID U8 – Enable/Disable U8 – Reserved channel count U8 – Maximum channels U16 – Constellation Option Flags)
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	7 + 6*N	0xA0	U16 – Maximum channels available U16 – Maximum channels to use U8 – Count of constellations to configure (N) N * (U8 – Constellation ID U8 – Enable/Disable U8 – Reserved channel count U8 – Maximum channels U16 – Constellation Option Flags)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*

Command GNSS Constellation Settings (use new settings)	**0x75**	**0x65**	**0x0C**	**0x12**	**0x12**	**0x21**	Function: **0x01** Max Ch: **0x0020** GNSS count: **0x02** 1st GNSS: ID (GPS): **0x00** Enable: **0x01** #Resrvd Ch: **0x08** Max #Ch: **0x10** Options: **0x0000** 2nd GNSS ID (GLNS): **0x06** Enable: **0x01** #Resrvd Ch: **0x08** Max #Ch: **0x0E** Options: **0x0000**	**0x84**	**0x5A**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x21** *Error code:***0x00**	**0x00**	**0xEC**

Copy-Paste version of the command: "7565 0C12 1221 0100 2002 0001 0810 0000 0601 080E 0000 845A"

GNSS SBAS Settings (0x0C, 0x22)

Description	This configures how SBAS satellites should be used for GNSS augmentation.		
Notes	*Function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *SBAS Enable:* 0x00 – Disable 0x01 – Enable *(default)* *Options Flags:* 0x0001 – Enable ranging *(default)* 0x0002 – Enable SBAS correction data *(default)* 0x0004 – Apply integrity information *For each satellite you wish to INCLUDE from SBAS corrections:* Satellite PRN# to include		
	Field Length	*Field Descriptor*	*Field Data*
Command	7 + 2*N	0x21	U8 – Function selector U8 – SBAS Enable/Disable U16 – SBAS Option Flags U8 – Count of Satellite PRN# to include (N) N * (U16 – Satellite PRN#)
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)

Reply field 2 (function = 2)	6 + 2*N	0xA1	U8 – SBAS Enable/Disable U16 – SBAS Option Flags U8 – Count of Satellite PRN# to include (N) N * (U16 – Satellite PRN#)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command GNSS SBAS Settings (use new settings)	**0x75**	**0x65**	**0x0C**	**0x0B**	**0x0B**	**0x22**	Function: **0x01** SBAS En: **0x01** Options: **0x0003** PRN Exc Cnt: **0x02** 1st PRN Exc: PRN# : **0x0078** 2nd PRN Exc: PRN# : **0x0079**	**0x16**	**0x5C**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x22** *Error code:* **0x00**	**0x01**	**0xEE**

Copy-Paste version of the command: “7565 0C0B 0B22 0101 0003 0200 7800 7916 5C”

Device Startup Settings (0x0C, 0x30)

Description	Save, Load, or Reset to Default the values for all device settings.		
Notes	*Possible function selector values:* 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Load factory default settings ***When a save current settings command is issued a brief data disturbance may occur.*		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x30	U8 –Function Selector
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Save All	**0x75**	**0x65**	**0x0C**	**0x03**	**0x03**	**0x30**	*Fctn(Save)*:**0x03**	**0x1F**	**0x45**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd*:**0x30** *Error code*:**0x00**	**0x0F**	**0x0A**

Copy-Paste version of the command: "7565 0C03 0330 031F 45"

Accel Bias (0x0C, 0x37)

Advanced

Description	Set the value, or read the current value of the IMU7 Accelerometer Bias Vector. For all functions except 0x01 and 0x06 (apply new settings), the new vector value is ignored. The bias value is subtracted from the scaled accelerometer value prior to output.		
Notes	*Possible function selector values:* 0x01 – Apply new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Load factory default settings 0x06 – Apply new settings with no ACK/NACK Reply		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x37	U8 – Function Selector float – X Accel Bias Value float – Y Accel Bias Value float – Z Accel Bias Value
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x9A	float – current X Accel Bias Value float – current Y Accel Bias Value float – current Z Accel Bias Value

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Accel Bias	**0x75**	**0x65**	**0x0C**	**0x0F**	**0x0F**	**0x37**	*Fctn(Apply):***0x01** *Field (Bias):* **0x00000000** **0x00000000** **0x00000000**	**0x3C**	**0x75**
Reply	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x37**	**0x16**	**0x18**

ACK/NACK							*Error code:***0x00**		

Copy-Paste version of the command: "7565 0C0F 0F37 0100 0000 0000 0000 0000 0000 003C 75"

Gyro Bias (0x0C, 0x38)

Advanced

Description	Set the value, or read the current value of the IMU7 Gyro Bias Vector. For all functions except 0x01 and 0x06 (apply new settings), the new vector value is ignored. The bias value is subtracted from the scaled Gyro value prior to output.		
Notes	*Possible function selector values:* 0x01 – Apply new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Load factory default settings 0x06 – Apply new settings with no ACK/NACK Reply		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x38	U8 – Function Selector float – X Gyro Bias Value float – Y Gyro Bias Value float – Z Gyro Bias Value
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x9B	float – current X Gyro Bias Value float – current Y Gyro Bias Value float – current Z Gyro Bias Value

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Gyro Bias	**0x75**	**0x65**	**0x0C**	**0x0F**	**0x0F**	**0x38**	*Fctn(Apply):***0x01** *Field (Bias):*	**0x3D**	**0x83**

							0x00000000 **0x00000000** **0x00000000**		
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x38** *Error code:***0x00**	**0x17**	**0x1A**

Copy-Paste version of the command: "7565 0C0F 0F38 0100 0000 0000 0000 0000 0000 003D 83"

Capture Gyro Bias (0x0C, 0x39)

Description	This command will cause the GQ4-45 to sample its sensors for the specified number of milliseconds. The resulting data will be used to initialize its orientation, and to estimate its gyro bias error. The estimated gyro bias error will be automatically written to the Gyro Bias vector. The bias vector is not saved as a startup value. If you wish to save this vector, use the Gyro Bias command.		
Notes	Possible Sampling Time values: Total sampling time in units of milliseconds. Range of values: 1000 to 30000. Note: The GQ4-45 must be stationary during the execution of the Capture Gyro Bias Operation.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x04	0x39	U16 – Sampling Time (milliseconds)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x9B	float – current X Gyro Bias Value float – current Y Gyro Bias Value float – current Z Gyro Bias Value

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0x39**	Sampling Time:	**0x5E**	**0xE0**

							0x2710		
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x12**	**0x04**	**0xF1**	Echo cmd: **0x39** Error code: **0x00**		
Reply field 2 Bias Vector					**0x0E**	**0x9B**	Field (Bias): **0x00000000** **0x00000000** **0x00000000**	**0xCF**	**0x19**

Copy-Paste version of the command: "7565 0C04 0439 2710 5EE0"

Magnetometer Hard Iron Offset (0x0C, 0x3A)

Description	This command will read or write values to the magnetometer Hard Iron Offset Vector. For all functions except 0x01 and 0x06 (apply new settings), the new vector value is ignored. The offset value is subtracted from the scaled Mag value prior to output.		
Notes	*Possible function selector values:* 0x01 – Apply new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Load factory default settings 0x06 – Apply new settings with no ACK/NACK Reply		
Notes	*Default values:* Hard Iron Offset: [0,0,0]		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x3A	U8 – Function Selector Float – X Hard Iron Offset Float – Y Hard Iron Offset Float – Z Hard Iron Offset
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x9C	Float – current X Hard Iron Offset Float – current Y Hard Iron Offset Float – current Z Hard Iron Offset

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*

Command *Hard Iron Offset*	**0x75**	**0x65**	**0x0C**	**0x0F**	**0x0F**	**0x3A**	*Fctn(Apply):***0x01** *Offset Vector:* **0x00000000** **0x00000000** **0x00000000**	**0x3F**	**0x9F**
Reply field 1 *ACK/NACK*	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x3A** *Error code:* **0x00**	**0x19**	**0x1** **E**

Copy-Paste version of the command: "7565 0C0F 0F3A 0100 0000 0000 0000 0000 0000 003F 9F"

Magnetometer Soft Iron Matrix (0x0C, 0x3B)

Description	This command will read or write values to the magnetometer Soft Iron Compensation Matrix. The values for this matrix are determined empirically by external software algorithms based on calibration data taken after the device is installed in its application. These values can be obtained and set by using the MicroStrain "GX Iron Calibration" application. The matrix is applied to the scaled magnetometer vector prior to output.		
Notes	*Possible function selector values:* 0x01 – Apply new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Load factory default settings 0x06 – Apply new settings with no ACK/NACK Reply *Default values:* Soft Iron Compensation Matrix (identity matrix; row order): [1,0,0][0,1,0][0,0,1]		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x27	0x3B	U8 – Function Selector float – $m_{1,1}$ float – $m_{1,2}$ float – $m_{1,3}$ float – $m_{2,1}$ float – $m_{2,2}$ float – $m_{2,3}$ float – $m_{3,1}$ float – $m_{3,2}$ float – $m_{3,3}$
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x26	0x9D	float – $m_{1,1}$ float – $m_{1,2}$ float – $m_{1,3}$ float – $m_{2,1}$ float – $m_{2,2}$ float – $m_{2,3}$ float – $m_{3,1}$ float – $m_{3,2}$ float – $m_{3,3}$

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*

Command *Soft Iron Matrix*	**0x75**	**0x65**	**0x0C**	**0x27**	**0x27**	**0x3B**	*Fctn(Apply):***0x01** *Comp Matrix:* **0x3F800000** **0x00000000** **0x00000000** **0x00000000** **0x3F800000** **0x00000000** **0x00000000** **0x00000000** **0x3F800000**	**0xA** **D**	**0x59**
Reply field 1 *ACK/NACK*	**0x75**	**0x65**	**0x0C**	**0x12**	**0x04**	**0xF1**	*Echo cmd:* **0x3B** *Error code:* **0x00**	**0x1** **A**	**0x20**

Copy-Paste version of the command: "7565 0C27 273B 013F 8000 0000 0000 0000 0000 0000 0000 003F 8000 0000 0000 0000 0000 0000 0000 003F 8000 00AD 59"

Coning and Sculling Enable (0x0C, 0x3E)

<table>
<tr><td>Description</td><td colspan="9">Set, read, or save the Coning and Sculling compensation Enable. This function sets the Coning and Sculling compensation Enable. For all functions except 0x01 (use new setting), the new parameter values are ignored.</td></tr>
<tr><td>Notes</td><td colspan="9">Possible function selector values:

0x01 – Apply new setting
0x02 – Read back current setting
0x03 – Save current settings as startup setting
0x04 – Load saved startup setting
0x05 – Load factory default setting

The enable flag can be either:

0x00 – disable the Coning and Sculling compensation.
0x01 – enable the Coning and Sculling compensation. (default)</td></tr>
<tr><td>Field Format</td><td colspan="2">Field Length</td><td colspan="3">Field Descriptor</td><td colspan="4">Field Data</td></tr>
<tr><td>Command</td><td colspan="2">0x10</td><td colspan="3">0x3E</td><td colspan="4">U8 – Function Selector
U8 – New Coning and Sculling enable setting</td></tr>
<tr><td>Reply field 1 ACK/NACK</td><td colspan="2">0x04</td><td colspan="3">0xF1</td><td colspan="4">U8 – echo the command descriptor
U8 – error code (0:ACK, not 0:NACK)</td></tr>
<tr><td>Reply field 2 (function = 2)</td><td colspan="2">0x03</td><td colspan="3">0x9E</td><td colspan="4">U8 – Current Coning and Sculling enable setting</td></tr>
<tr><td rowspan="2">Example</td><td colspan="4">MIP Packet Header</td><td colspan="3">Fields</td><td colspan="2">Checksum</td></tr>
<tr><td>Sync1</td><td>Sync2</td><td>Desc Set</td><td>Payload Length</td><td>Field Length</td><td>Field Desc.</td><td>Field Data</td><td>MSB</td><td>LSB</td></tr>
<tr><td>Command Enable Settings</td><td>0x75</td><td>0x65</td><td>0x0C</td><td>0x04</td><td>0x04</td><td>0x3E</td><td>Fctn (Apply): 0x01
Enable: 0x01</td><td>2E</td><td>94</td></tr>
<tr><td>Reply ACK/NACK</td><td>0x75</td><td>0x65</td><td>0x0C</td><td>0x04</td><td>0x04</td><td>0xF1</td><td>Echo cmd: 0x3E
Error code:0x00</td><td>1D</td><td>26</td></tr>
</table>

Copy-Paste version of the command: “7565 0C04 043E 0101 2E94”

UART BAUD Rate (0x0C, 0x40)

Description	Change, read, or save the BAUD rate of the main communication channel (UART1).For all functions except 0x01 (use new settings), the new BAUD rate value is ignored.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Supported BAUD rates are:* 9600, 19200, 115200*(default)*, 230400, 460800 ⚠ The ACK/NACK packet is sent at the *current* baud rate and then there is a 0.25 second delay before the device will respond to commands at the *new* BAUD rate.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x07	0x40	U8 – Function Selector U32 –New BAUD rate
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x06	0x87	U32 – Current BAUD rate

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command Set BAUD Rate Command	**0x75**	**0x65**	**0x0C**	**0x07**	**0x07**	**0x40**	*Fctn(USE)*:**0x01** *BAUD (115200)*: **0x0001C200**	**0xF8**	**0xDA**
Reply	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd*:**0x40**	**0x1F**	**0x2A**

ACK/NACK							*Error code:***0x00**		

Copy-Paste version of the command: "7565 0C07 0740 0100 01C2 00F8 DA"

Advanced Low-Pass Filter Settings (0x0C, 0x50)

Description	Advanced configuration for low-pass filter settings. The scaled data quantities are **by default** filtered through a single-pole IIR low-pass filter which is configured with a -3dB cutoff frequency of half the reporting frequency (set by decimation factor in the IMU Message Format command) to prevent aliasing on a per data quantity basis. This advanced configuration command allows for the cutoff frequency to be configured independently of the data reporting frequency as well as allowing for a complete bypass of the digital low-pass filter for all scaled data quantities.
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Possible data descriptors:* 0x04 – Scaled accel data 0x05 – Scaled gyro data 0x06 – Scaled mag data 0x17 – Scaled pressure data *Possible filter enable values:* 0x01 – Apply low-pass filter 0x00 – Do not apply low-pass filter *Manual filter bandwidth configuration:* 0x01 – Use user specified -3 dB cutoff frequency 0x00 – Automatically configure -3 dB cutoff frequency to half reporting rate

	-3 dB Cutoff Frequency: Cutoff Frequency value specified must be no greater than 250 Hz. ***This value in a write command is ignored if Automatic Bandwidth is selected.* *Reserved Byte:* This byte is reserved for internal use and should be left in the 0x00 state		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x09	0x50	U8 – Function Selector U8 – Data Descriptor U8 – Low-Pass Filter Enable/Disable U8 – Manual/Auto -3 dB Cutoff Frequency Configuration U16 – -3 dB Cutoff Frequency U8 – Reserved Byte
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x08	0x8B	U8 – Data Descriptor U8 – Filter (0x01: Enabled, 0x00: Disabled) U8 – Cutoff Frequency (0x00: Auto, 0x01: Manual) U16 – -3 dB Cutoff Frequency Hz U8 – Reserved

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command IMU Message Format (use new settings)	**0x75**	**0x65**	**0x0C**	**0x09**	**0x09**	**0x50**	Function: **0x01** Scaled Accel: **0x04** Enable Filter: **0x01** Automatic Cutoff Configuration: **0x00** -3 dB Cutoff	**0x4C**	**0x6D**

							Frequency: **0x0000** (ignored for automatic cutoff configuration) Reserved: **0x00**		
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x50** *Error code:***0x00**	**0x2F**	**0x4A**

Copy-Paste version of the commands: "7565 0C09 0950 0104 0100 0000 004E 80"

Complementary Filter Settings (0x0C, 0x51)

Description	Configuration for the AHRS complementary filter. The Complementary Filter data outputs are supported in the IMU/AHRS Data set (0x80) to provide compatibility with the 3DM-GX3.		
Notes	Possible function selector values: 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Possible up/north compensation enable values: 0x00 – Disable 0x01 – Enable (default) Range of up/north compensation time constants: 1-1000 seconds, default = 10 seconds Values outside of the specified range for up/north compensation will be NACK'd. ⚠ The Complementary Filter provides attitude outputs (Matrix, Euler, Quaternion, Up, and North) that are independent of the Estimation Filter outputs. The CF outputs are calculated using the same algorithm as the 3DM-GX3 series of Inertial Devices. This provides drop-in compatibility that duplicates the performance of the 3DM-GX3. It is highly recommended that you transition to the EF outputs as they will provide better performance as well as compatibility with higher grade devices such as the 3DM-RQ1.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0D	0x51	U8 – Function selector U8 – Up compensation enable U8 – North compensation enable float – Up compensation time constant (sec) float – North compensation time constant (sec)

Reply *ACK/NACK*	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 *(function = 2)*	0x0C	0x97	U8 – Up compensation enable U8 – North compensation enable float – Up compensation time constant (sec) float – North compensation time constant (sec)

Examples	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command *IMU Message Format (use new settings)*	0x75	0x65	0x0C	0x0D	0x0D	0x51	Function Selector: 0x01 (Write) Up Compensation Enable: 0x01 (enable) North Compensation Enable: 0x01 (enable) Up Compensation Time Constant: 5.0 (sec) North Compensation Time Constant: 5.0 (sec)	0xXX	0xXX
Reply *ACK/NACK*	0x75	0x65	0x0C	0x04	0x04	0xF1	*Echo cmd:*0x51 *Error code:*0x00	0x	0x

Copy-Paste version of the commands: "7565 0C09 0951 0104 0100 0000 00"

Device Status (0x0C, 0x64)

Description	Get the device-specific status for the 3DM-GQ4-45
Notes	Reply has two fields: “ACK/NACK” and “Device Status Field”. The device status field may be one of two selectable formats – basic and diagnostic. The reply data for this command is device specific. The reply is specified by two parameters in the command. The first parameter is the model number (which for the 3DM-GQ4-45 is always = 6250 (0x186A)). That is followed by a status selector byte which determines the type of data structure returned. In the case of the 3DM-GQ4-45, there are two selector values – one to return a basic status structure and a second to return an extensive diagnostics status structure. A list of available values for the selector values and specific fields in the data structure are as follows: *Possible Status Selector Values:* 0x01 – Basic Status Structure 0x02 – Diagnostic Status Structure *Possible Communication Mode Values:* 0x01 – Standard MIP Mode 0x02 – Advanced Sensor Direct Mode 0x03 – Advanced GNSS Direct Mode *Possible Communication Device Values:* 0x01 - Com1 (Serial) *Possible System State Values:* 0x0001 – Initialization 0x0002 – Sensor Start-up 0x0003 – Running *Possible Com1 State, GNSS Driver State, GNSS Port State, IMU Driver State, IMU Port State Values:* 0x00 – Not Initialized

	0x01 – Initialized					
Field Format	*Field Length*	*Field Descriptor*	*Field Data*			
Command	0x02	0x64	U16-Device Model Number: set = 6250 (0x186A) U8-Status Selector			
Reply field 1 ACK/NACK Field	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)			
Reply field 2 Basic Device Status Field	0x0F	0x90	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	Echo of the Device Model Number	U16	N/A
			2	Echo of the selector byte	U8	N/A
			3	Status Flags (Reserved)	U32	N/A
			7	System state	U16	See Notes
			9	System Timer (since start-up)	U32	millisecond s
Reply field 2 Diagnostic Device Status Field	0x56	0x90	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	Echo of the Device Model Number	U16	N/A
				Echo of the selector byte	U8	N/A
				Status Flags (Reserved)	U32	N/A
				System state	U16	See Notes
				System Timer (since start-up)	U32	millisecond s
				GNSS Power On	U8	1 – on 0 – off
				Number of GNSS 1PPS Pulses	U32	Count
				Last GNSS 1PPS (System Timer)	U32	millisecond s
				IMU Stream Enabled	U8	1 – on 0 – off
				GNSS Stream Enabled	U8	1 – on 0 – off
				Estimation Filter Stream Enabled	U8	1 – on 0 – off
				Outgoing IMU Stream Dropped Packet Count	U32	Count

				Outgoing GNSS Stream Dropped Packet Count	U32	Count
				Outgoing Estimation Filter Stream Dropped Packet Count	U32	Count
				Number of bytes written to com port	U32	Count
				Number of bytes read from com port	U32	Count
				Number of overruns when writing to com port	U32	Count
				Number of overruns when reading from com port	U32	Count
				Number of IMU message parsing errors	U32	Count
				Total IMU messages read	U32	Count
				Last IMU message read (System Timer)	U32	Millisecond
				Number of GNSS message parsing errors	U32	Count
				Total GNSS messages read	U32	Count
				Last GNSS message read (System Timer)	U32	Millisecond
				Magnetometer initialization failed	U8	1 – fail 0 – OK
				Pressure sensor initialization failed	U8	1 – fail 0 – OK
				GNSS receiver initialization failed	U8	1 – fail 0 – OK

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	MSB	LSB
Command Get Device Status (return Basic Status structure: selector = 1)	**0x75**	**0x65**	**0x0C**	**0x05**	**0x05**	**0x64**	Model # (6250): **0x186A** Status Selector (basic status): **0x01**	**0xD7**	**0x7D**
Reply field 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x15**	**0x04**	**0xF1**	*Echo cmd:* **0x64** *Error code:* **0x00**		

Reply field 2 Device Status (Basic Status structure)					**0x0D**	**0x90**	Echo Model#: **0x186A** Echo Selector: **0x01** .... Additional Data	**0x##**	**0x##**

Copy-Paste version of the command: "7565 0C05 0564 186A 01D7 7D"

Raw RTCM 2.3 Message (0x0C, 0x20)

Description	Send Raw RTCM 2.3 differential corrections to the device		
Notes	The device currently only accepts message types 1, 2, 3, and 9. RTK corrections are not supported. The RTCM data can be limited to 1 message per command or streamed into the device by dividing the stream into reasonably-sized chunks (no greater than 253 bytes.)		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x2 + Num_bytes	0x20	Num_bytes of the raw RTCM2.3 stream
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0C**	**2+Num**	**2+Num**	**0x20**	*Num_Bytes of raw RTCM2.3 data*	**0xXX**	**0xXX**
Reply ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x20** *Error code:* **0x00**	**0xFF**	**0xEA**

Copy-Paste version of the command: N/A

Estimation Filter Commands

The Estimation Filter command set is specific to MicroStrain Inertial Navigation and advanced AHRS sensors.

Reset Filter (0x0D, 0x01)

Description	Reset the filter to the initialize state.		
Notes	If the auto-initialization feature is disabled, the initial attitude or heading must be set in order to enter the run state after a reset.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x01	N/A
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, non-zero:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x02**	**0x02**	**0x01**		**0xEC**	**0xF6**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x01** *Error code:* **0x00**	**0xE1**	**0xB2**

Copy-Paste version of the command: "7565 0D02 0201 ECF6"

Set Initial Attitude (0x0D, 0x02)

Description	Set the initial attitude.		
Notes	This command can only be issued in the "INIT" state and should be used with a good estimate of the vehicle attitude. The Euler Angles are the sensor body frame with respect to the local NED frame. The valid input ranges are as follows: Roll: [-π, π] Pitch: $[-\frac{\pi}{2}, \frac{\pi}{2}]$ Yaw: [-π, π]		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0E	0x02	Float – Roll (radians) Float – Pitch (radians) Float – Heading (radians)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0E**	**0E**	**02**	*Roll:*0x00000000 **(0.0f)** *Pitch:*0x00000000 **(0.0f)** *Heading:*0x00000000 **(0.0f)**	**0x05**	**0x6F**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x02** *Error code:* **0x00**	**0xE2**	**0xB4**

Copy-Paste version of the command: "7565 0D0E0E02 0000 0000 0000 0000 0000 0000 0000 0000 056F"

Set Initial Heading (0x0D, 0x03)

Description	Set the initial heading angle.		
Notes	This command can only be issued in the "INIT" state and should be used with a good estimation of Heading. The device will use this value in conjunction with the output of the accelerometers to determine the initial attitude estimate. The Euler Angles are the sensor body frame with respect to the local NED frame. The valid input range for heading is [-π, π].		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x06	0x03	Float – Heading (radians)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x06**	**0x06**	**0x03**	*Heading:*0x00000000 **(0.0f)**	**0xF6**	**0xE4**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x03** *Error code:* **0x00**	**0xE3**	**0xB6**

Copy-Paste version of the command: "7565 0D06 0603 0000 0000 F6E4"

Vehicle Dynamics Mode (0x0D, 0x10)

Description	Set, read, or save the vehicle dynamics mode. For all functions except 0x01 (use new settings), the new dynamics mode value is ignored.
Notes	*Possible function selector values:* 0x01 – Apply new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Load factory default settings *Possible Modes:*

Mode	Use	Altitude Limits	Velocity Limits
0x01 – Portable (default)	Applications with low acceleration	12,000 m	Horizontal - 310 m/s Vertical - 50 m/s
0x02 – Automotive	Low vertical acceleration, wheeled-vehicle dynamics	6000 m	Horizontal - 84 m/s Vertical - 15 m/s
0x03 – Airborne	Airborne application up to 2 Gs	50,000 m	Horizontal - 250 m/s Vertical - 100 m/s
0x04 – Airborne (High G)	Airborne application up to 4 Gs	50,000 m	Horizontal - 250 m/s Vertical - 100 m/s

Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x04	0x10	U8 – Function Selector U8 – New Dynamics Mode
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	3	0x80	U8 –Current Dynamics Mode

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command DynamicsMode	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0x10**	*Fctn (Apply):* **0x01** *Mode*	**0x01**	**0x10**

							(Portable):**0x01**		
Reply *ACK/NACK*	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x10** *Error code:***0x00**	**0xF0**	**0xD0**

Copy-Paste version of the command: "7565 0C04 0410 01010110"

Sensor to Vehicle Frame Transformation (0x0D, 0x11)

Description	Set the sensor to vehicle frame transformation matrix using Roll, Pitch, and Yaw Euler angles. These angles define the rotation *from* the sensor body frame *to* the fixed vehicle frame. Please reference the device Theory of Operation for more information.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings This transformation affects the following output quantities: IMU: Scaled Acceleration Scaled Gyro Delta Theta Delta Velocity Estimation Filter: Estimated Orientation, Quaternion Estimated Orientation, Matrix Estimated Orientation, Euler Angles Estimated Linear Acceleration Estimated Angular Rate Estimated Gravity Vector		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*

Command	0x0F	0x11	U8 – Function Selector Float – Roll Angle (radians) Float – Pitch Angle (radians) Float – Yaw Angle (radians)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x81	Float – Roll Angle (radians) Float – Pitch Angle (radians) Float – Yaw Angle (radians)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x0F**	**0x11**	*Fctn (Apply):* **0x01** *Roll*:0x00000000 **(0.0f)** *Pitch*:0x00000000 **(0.0f)** *Yaw*:0x00000000 **(0.0f)**	**0x17**	**0x72**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x11** *Error code*:**0x00**	**0xF1**	**0xD2**

Copy-Paste version of the command: "7565 0D0F 0F11 0100 0000 0000 0000 0000 0000 0017 72"

Sensor to Vehicle Frame Offset (0x0D, 0x12)

Description	Set the sensor to vehicle frame offset, expressed in the sensor frame. Please reference the device Theory of Operation for more information.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings This offset affects the following output quantities: Estimated LLH Position The offset vector magnitude is limited to 10 meters.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x12	U8 – Function Selector Float – X (meters, sensor body frame) Float – Y (meters, sensor body frame) Float – Z (meters, sensor body frame)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x82	Float – X (meters, sensor body frame) Float – Y (meters, sensor body frame) Float – Z (meters, sensor body frame)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x0F**	**0x12**	*Fctn (Apply):* **0x01** *X:*0x00000000 **(0.0f)** *Y:*0x00000000 **(0.0f)** *Z:*0x00000000	**0x18**	**0x80**

							(0.0f)		
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x12** *Error code:* **0x00**	**0xF2**	**0xD4**

Copy-Paste version of the command: "7565 0D0F 0F12 0100 0000 0000 0000 0000 0000 0018 80"

Antenna Offset (0x0D, 0x13)

Description	Set the sensor to antenna offset, expressed in the sensor frame from the sensor inertial origin to the GNSS antenna RF center.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings The offset vector magnitude is limited to 10 meters.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x13	U8 – Function Selector Float – X (meters, sensor body frame) Float – Y (meters, sensor body frame) Float – Z (meters, sensor body frame)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x83	Float – X (meters, sensor body frame) Float – Y (meters, sensor body frame) Float – Z (meters, sensor body frame)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x0F**	**0x13**	*Fctn (Apply):* **0x01**	**0x19**	**0x8E**

							X:0x00000000 **(0.0f)** *Y*:0x00000000 **(0.0f)** *Z*:0x00000000 **(0.0f)**		
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x13** *Error code:***0x00**	**0xF3**	**0xD6**

Copy-Paste version of the command: "7565 0D0F 0F13 0100 0000 0000 0000 0000 0000 0019 8E"

Estimation Control Flags (0x0D, 0x14)

Description	Controls which parameters are estimated by the Kalman Filter.
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Available Flags :* 0x0001 – Enable Gyro Bias Estimation (Recommended) 0x0002 – Enable Accel Bias Estimation (Recommended) 0x0004 – Enable Gyro Scale Factor Estimation (Optional) 0x0008 – Enable Accel Scale Factor Estimation (Optional) 0x0010 – Enable GNSS Antenna Offset Estimation (Optional) *Examples :* 0xFFFF – Enable all 0xFFF5 – Disable Accel Scale Factor and Bias Estimation 0xFFFA – Disable Gyro Scale Factor and Bias Estimation 0xFFEF – Disable GNSS Antenna Offset Estimation (note: Any bit without a designated function should be set to 1 for future compatibility.)

Field Format	Field Length	Field Descriptor	Field Data
Command	0x05	0x14	U8 – Function Selector U16 – Estimation Control Flags
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x04	0x84	U16 – Estimation Control Flags

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x05**	**0x05**	**0x14**	*Fctn (Apply):* **0x01** *Flags:*0xFFFF **(Enable all states)**	**0x04**	**0x27**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x14** *Error code:***0x00**	**0xF4**	**0xD8**

Copy-Paste version of the command: "7565 0D05 0514 01FF FF04 27"

GNSS Source Control (0x0D, 0x15)

Description	Control the source of GNSS information used to update the Kalman Filter.
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Possible GNSS Source values:*

	0x01 – Internal GNSS 0x02 – External GNSS (Requires user to provide GNSS information via the "External GNSS Update" command) Changing the GNSS source while the sensor is in the "running" state will temporarily place it back in the "init" state until the new source of GNSS data is received.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x04	0x15	U8 – Function Selector U8 –GNSS Source
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x03	0x86	U8 – GNSS Source

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0x15**	*Fctn (Apply):* **0x01** *Source:*0x02 **(External GNSS)**	**0x07**	**0x20**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x15** *Error code:***0x00**	**0xF5**	**0xDA**

Copy-Paste version of the command: "7565 0D04 0415 0102 0720"

External GNSS Update (0x0D, 0x16)

Description	Trigger a filter update step using external GNSS information.
Notes	GNSS source control must be set to external for this command to update the filter; it will be ignored/NACK'd otherwise. The maximum rate for this message is 20 Hz.

Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x48	0x16	Double – GPS Time of Week (seconds) U16 – GPS Week Number Double – Latitude (deg) Double – Longitude (deg) Double – Altitude above WGS84 Ellipsoid (m) Float – North Velocity (m/s) Float – East Velocity (m/s) Float – Down Velocity (m/s) Float – North Position Uncertainty (m, 1-sigma) Float – East Position Uncertainty (m, 1-sigma) Float – Down Position Uncertainty (m, 1-sigma) Float – North Velocity Uncertainty (m/s, 1-sigma) Float – East Velocity Uncertainty (m/s, 1-sigma) Float – Down Velocity Uncertainty (m/s, 1-sigma)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**48**	**48**	**16**	*GPS Tow:* **0.0d** *GPS Week: 0x0000* *Latitude:***0.0d** *Longitude:***0.0d** *Height:***0.0d** *Vel North:***0.0f** *Vel East:***0.0f** *Vel Down:***0.0f** *Pos Sigma (N)***0.5f** *Pos Sigma (E)***0.5f** *Pos Sigma (D)***0.5f** *Vel Sigma (N)***0.1f** *Vel Sigma (E)***0.1f** *Vel Sigma (D)***0.1f**	**0xXX**	**0xXX**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x16** *Error code:* **0x00**	**0xF6**	**0xDC**

Copy-Paste version of the command: N/A

External Heading Update (0x0D, 0x17)

Description	Trigger a filter update step using external heading information ***The heading must be the sensor frame with respect to the NED frame.***
Notes	The heading update control must be set to external for this command to update the filter; it will be ignored/NACK'd otherwise. The maximum rate for this message is 20 Hz. Angle uncertainties of 0.0 will be NACK'd. Possible Heading Type Commands: 0x01 – True Heading* 0x02 – Magnetic Heading* ***Note:*** - On the -25 model, if the declination source (0x0D, 0x43) is not valid, ***true heading updates*** will be NACK'd. - On the -45 model, if the declination source is invalid, ***magnetic heading updates*** will be NACK'd.

Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0B	0x17	Float – Heading Angle (radians, true north, +-PI) Float – Heading Angle Uncertainty (radians, 1-sigma) U8 – Heading type (1 – true, 2 – magnetic)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*

Command	**0x75**	**0x65**	**0x0D**	**0x0B**	**0x0B**	**0x17**	*Angle:***0.0f** *Angle Sigma:***0.01f** Heading Type: **0x01**(True)	**0xXX**	**0xXX**
Reply *ACK/NACK*	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x17** *Error code:* **0x00**	**0xF7**	**0xDE**

Copy-Paste version of the command: N/A

Heading Update Control (0x0D, 0x18)

<table>
<tr><td>Description</td><td colspan="3">Select the source for heading updates to the Kalman Filter.</td></tr>
<tr><td>Notes</td><td colspan="3">Possible function selector values:
0x01 – Use new settings
0x02 – Read back current settings.
0x03 – Save current settings as startup settings
0x04 – Load saved startup settings
0x05 – Reset to factory default settings

Possible Enable Flag values:
0x00 – Disable Heading Updates
0x01 – Use the Internal Magnetometer for Heading Updates
0x02 – Use the Internal GNSS Velocity Vector for Heading Updates**
0x03 – Use external heading updates

**To use the Internal GNSS velocity vector for heading updates, the target application must have no (or minimal) side-slip; this is true in most ground vehicle applications. Additionally, when this option is selected, the X-axis of the sensor must be co-aligned with the direction of travel of the vehicle.</td></tr>
<tr><td>Field Format</td><td>Field Length</td><td>Field Descriptor</td><td>Field Data</td></tr>
<tr><td>Command</td><td>0x04</td><td>0x18</td><td>U8 – Function Selector
U8 –Enable Flag</td></tr>
<tr><td>Reply ACK/NACK</td><td>0x04</td><td>0xF1</td><td>U8 – echo the command descriptor
U8 – error code (0:ACK, not 0:NACK)</td></tr>
<tr><td>Reply field 2 (function = 2)</td><td>0x03</td><td>0x87</td><td>U8 – Enable Flag</td></tr>
</table>

<table>
<tr><td rowspan="2">Example</td><td colspan="4">MIP Packet Header</td><td colspan="3">Fields</td><td colspan="2">Checksum</td></tr>
<tr><td>Sync1</td><td>Sync2</td><td>Desc Set</td><td>Payload Length</td><td>Field Length</td><td>Field Desc.</td><td>Field Data</td><td>MSB</td><td>LSB</td></tr>
<tr><td>Command</td><td>0x75</td><td>0x65</td><td>0x0D</td><td>0x04</td><td>0x04</td><td>0x18</td><td>Fctn (Apply): 0x01
Enable:0x01
(Enable Mag. Updates)</td><td>0x09</td><td>0x28</td></tr>
</table>

Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x18** *Error code:***0x00**	**0xF8**	**0xE0**

Copy-Paste version of the command: "7565 0D04 0418 0101 0928"

Auto-Initialization Control (0x0D, 0x19)

Description	Enable/Disable automatic initialization upon device startup.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Possible Enable Flag values:* 0x00 – Disable auto-initialization 0x01 – Enable auto-initialization (requires valid heading source)		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x04	0x19	U8 – Function Selector U8 – Enable Flag
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x03	0x88	U8 – Enable Flag

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0x19**	*Fctn (Apply):* **0x01** *Enable:*0x01 **(Enable auto-initialization)**	**0x0A**	**0x2B**

Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x19** *Error code:***0x00**	**0xF9**	**0xE2**

Copy-Paste version of the command: "7565 0D04 0419 0101 0A2B"

Accelerometer Noise Standard Deviation (0x0D, 0x1A)

Description	Set the expected accelerometer noise 1-sigma values. This function can be used to tune the filter performance in the target application.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Each of the noise values must be greater than 0.0 The noise value represents process noise in the GX3-45 NAV Estimation Filter. Changing this value modifies how the filter responds to dynamic input and can be used to tune the performance of the filter. Default values provide good performance for most laboratory conditions.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x1A	U8 – Function Selector Float – X Accel Noise 1-sigma (meters/second^2) Float – Y Accel Noise 1-sigma (meters/second^2) Float – Z Accel Noise 1-sigma (meters/second^2)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x89	Float – X Accel Noise 1-sigma (meters/second^2) Float – Y Accel Noise 1-sigma

			(meters/second^2) Float – Z Accel Noise 1-sigma (meters/second^2)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x0F**	**0x1A**	*Fctn (Apply):* **0x01** *X:*(**0.02f**) *Y:*(**0.02f**) *Z:*(**0.02f**)	**0x60**	**0xA3**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x1A** *Error code:***0x00**	**0xFA**	**0xE4**

Copy-Paste version of the command: "7565 0D0F 0F01 1A013CA3D70A3CA3D70A3CA3D760A3"

Gyroscope Noise Standard Deviation (0x0D, 0x1B)

Description	Set the expected gyroscope noise 1-sigma values. This function can be used to tune the filter performance in the target application.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Each of the noise values must be greater than 0.0 The noise value represents process noise in the GX3-45 NAV Estimation Filter. Changing this value modifies how the filter responds to dynamic input and can be used to tune the performance of the filter. Default values provide good performance for most laboratory conditions.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x1B	U8 – Function Selector Float – X Gyro Noise 1-sigma (rad/second) Float – Y Gyro Noise 1-sigma (rad/second) Float – Z Gyro Noise 1-sigma (rad/second)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0x8A	Float – X Gyro Noise 1-sigma (rad/second) Float – Y Gyro Noise 1-sigma (rad/second) Float – Z Gyro Noise 1-sigma (rad/second)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x0F**	**0x1B**	*Fctn (Apply):* **0x01** *X:*(**0.0000539f)** *Y:*(**0.0000539f)** *Z:*(**0.0000539f)**	**0xDE**	**0xE8**

Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x1B** *Error code:***0x00**	**0xFB**	**0xE6**

Copy-Paste version of the command: "7565 0D0F 0F1B 013A 0D4B AD3A 0D4B AD3A 0D4B ADDE E8"

Magnetometer Noise Standard Deviation (0x0D, 0x42)

Description	Set the expected magnetometer noise 1-sigma values. This function can be used to tune the filter performance in the target application.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Each of the noise values must be greater than 0.0 The noise value represents process noise in the GX3-45 NAV Estimation Filter. Changing this value modifies how the filter responds to dynamic input and can be used to tune the performance of the filter. Default values provide good performance for most laboratory conditions.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x0F	0x42	U8 – Function Selector Float – X Mag Noise 1-sigma (gauss) Float – Y Mag Noise 1-sigma (gauss) Float – Z Mag Noise 1-sigma (gauss)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x0E	0xB1	Float – X Mag Noise 1-sigma (gauss) Float – Y Mag Noise 1-sigma (gauss) Float – Z Mag Noise 1-sigma (gauss)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x0F**	**0x42**	*Fctn (Apply):* **0x01** *X*:(**0.02f)** *Y*:(**0.02f)** *Z*:(**0.02f)**	**0x**	**0x**
Reply *ACK/NACK*	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x42** *Error code:***0x00**	**0x22**	**0x34**

Copy-Paste version of the command: “”

Accelerometer Bias Model Parameters (0x0D, 0x1C)

Description	Set the accelerometer bias model parameters.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Each of the noise values must be greater than 0.0		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x1B	0x1C	U8 – Function Selector Float – X Accel Bias Beta (1/second) Float – Y Accel Bias Beta (1/second) Float – Z Accel Bias Beta (1/second) Float – X Accel Bias Noise 1-sigma (m/s^2) Float – Y Accel Bias Noise 1-sigma (m/s^2) Float – Z Accel Bias Noise 1-sigma (m/s^2)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x1A	0x8B	Float – X Accel Bias Beta (1/second) Float – Y Accel Bias Beta (1/second) Float – Z Accel Bias Beta (1/second) Float – X Accel Bias Noise 1-sigma (m/s^2) Float – Y Accel Bias Noise 1-sigma (m/s^2) Float – Z Accel Bias Noise 1-sigma (m/s^2)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x1B**	**0x1C**	*Fctn (Apply):* **0x01** *X Beta:* (**0.01f**) *Y Beta:* (**0.01f**) *Z Beta:* (**0.01f**) *X Noise:* (**0.0006f**) *Y Noise:* (**0.0006f**)	**0xXX**	**0xXX**

							Z Noise: **(0.0006f)**		
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x1C** *Error code:* **0x00**	**0xFC**	**0xE8**

Copy-Paste version of the command: N/A

Gyroscope Bias Model Parameters (0x0D, 0x1D)

Description	Set the gyroscope bias model parameters.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Each of the noise values must be greater than 0.0		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x1B	0x1D	U8 – Function Selector Float – X Gyro Bias Beta (1/second) Float – Y Gyro Bias Beta (1/second) Float – Z Gyro Bias Beta (1/second) Float – X Gyro Bias Noise 1-sigma (rad /second) Float – Y Gyro Bias Noise 1-sigma (rad /second) Float – Z Gyro Bias Noise 1-sigma (rad /second)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x1A	0x8C	Float – X Gyro Bias Beta (1/second) Float – Y Gyro Bias Beta (1/second) Float – Z Gyro Bias Beta (1/second) Float – X Gyro Bias Noise 1-sigma (rad /second)

			Float – Y Gyro Bias Noise 1-sigma (rad /second) Float – Z Gyro Bias Noise 1-sigma (rad /second)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x0F**	**0x1B**	**0x1D**	*Fctn (Apply):* **0x01** *X Beta:* **(0.01f)** *Y Beta:* **(0.01f)** *Z Beta:* **(0.01f)** *X Noise:* **(0.00016f)** *Y Noise:* **(0. 00016f)** *Z Noise:* **(0. 00016f)**	**0xXX**	**0xXX**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x1D** *Error code:***0x00**	**0xFD**	**0xEA**

Copy-Paste version of the command: N/A

Zero Velocity Update (ZUPT) Control (0x0D, 0x1E)

Description	Control the use of zero velocity updates.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings The ZUPT is triggered when the scalar magnitude of the GNSS reported velocity vector is equal-to or less than the threshold value. The device will NACK threshold values that are less than zero (i.e. negative.)		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x08	0x1E	U8 – Function Selector

			U8 –Enable Value Float – ZUPT threshold (m/s)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x07	0x8D	U8– Enable Value Float – ZUPT threshold (m/s)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x08**	**0x08**	**0x1E**	*Fctn (Apply):* **0x01** *Enable:*0x01 **(Enable ZUPTs)** Threshold: 0x00000000 **(0.0f)**	**0x17**	**0xBA**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x1E** *Error code:***0x00**	**0xFE**	**0xEC**

Copy-Paste version of the command: "7565 0D08 081E 0101 00000000 17BA"

External Heading Update with Timestamp (0x0D, 0x1F)

Description	Trigger a filter update step using external heading information that is time-tagged with a specific GPS time. ***The heading must be the sensor frame with respect to the NED frame.***		
Notes	This is more accurate than the External Heading Update (0x0D, 0x17) and should be used in applications where the vehicle heading experiences high angular rate, which may cause significant error in the applied measurement due to the sampling, transmission, and processing time required for the command. Accurate time-stamping of the heading information is important. The maximum rate for this message is 20 Hz. Angle uncertainties of 0.0 will be NACK'd. Possible Heading Type Commands: 0x01 – True Heading* 0x02 – Magnetic Heading* ***Note*:** - On the -25 model, if the declination source (0x0D, 0x43) is not valid, ***true heading updates*** will be NACK'd. - On the -45 model, if the declination source is invalid, ***magnetic heading updates*** will be NACK'd.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x15	0x1F	Double – GPS TOW (time-of-week, seconds) U16 – GPS week number Float – Heading Angle (radians, true north, +- PI) Float – Heading Angle Uncertainty (radians, 1-sigma) U8 – Heading type (1 – true, 2 – magnetic)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, not 0:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x15**	**0x15**	**0x1F**	*GPS TOW:* ***30,000.0*** *GPS Week*	**0xXX**	**0xXX**

							Number: ***1700*** *Angle:*0.0f *Angle Sigma:*0.01f Heading Type: **0x01**(True)		
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x1F** *Error code:* **0x00**	**0xFF**	**0xEE**

Copy-Paste version of the command: N/A

Zero Angular Rate Update Control (0x0D, 0x20)

Description	Control the use of zero angular rate updates.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings The zero angular rate update is triggered when the scalar magnitude of the angular rate vector is equal-to or less than the threshold value. The device will NACK threshold values that are less than zero (i.e. negative.)		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x08	0x20	U8 – Function Selector U8 –Enable Value (0 – disable, 1 – enable) Float –Threshold (rad/s)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x07	0x8E	U8– Enable Value Float – ZUPT threshold (rad/s)
Example	*MIP Packet Header*	*Fields*	*Checksum*

	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x08**	**0x08**	**0x20**	*Fctn (Apply):* **0x01** *Enable:*0x01 **(Enable)** Threshold: 0x00000000 **(0.0f)**	**0x19**	**0xC8**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x20** *Error code:***0x00**	**0x00**	**0xF0**

Copy-Paste version of the command: "7565 0D08 0820 0101 00000000 19C8"

Tare Orientation (0x0D, 0x21)

Description	This function uses the current device orientation relative to the NED frame as the current sensor to vehicle transformation. This command is provided as a convenient way to set the sensor to vehicle frame transformation.
Notes	*Possible function selector values:* 0x01 – Use new settings 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Possible axis bitfield values:* 0x00 – Reset all axis 0x01 – Tare the roll axis 0x02 – Tare the pitch axis 0x04 – Tare the yaw axis Example Combinations: 0x03 – Tare the roll and pitch axis 0x07 – Tare all 3 axis Note: The filter must be in the "running" state to use this command.

Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x04	0x21	U8 – Function Selector U8 – Tare Axis Bitfield
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0x21**	*Fctn (Apply):* **0x01** *X*:0x07 (**All axis)**	**0x18**	**0x49**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x21** *Error code:***0x00**	**0x**	**0x**

Copy-Paste version of the command: "7565 0D04 0421 0107 1849"

Commanded Zero-Velocity Update (0x0D, 0x22)

Description	Perform a commanded ZUPT. The maximum rate for this message is 20 Hz.		
Notes			
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x22	N/A
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, non-zero:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x02**	**0x02**	**0x22**		**0x0D**	**0x17**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x22** *Error code:* **0x00**	**0x02**	**0xF4**

Copy-Paste version of the command: "7565 0D02 0222 0D17"

Commanded Zero-Angular Rate Update (0x0D, 0x23)

Description	Perform a commanded zero-angular rate update. The maximum rate for this message is 10 Hz.		
Notes			
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x02	0x23	N/A
Reply ACK/NACK	0x04	0xF1	U8 – echo the command byte U8 – error code (0:ACK, non-zero:NACK)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x02**	**0x02**	**0x23**		**0x0E**	**0x18**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x23** *Error code:* **0x00**	**0x03**	**0xF6**

Copy-Paste version of the command: "7565 0D02 0222 0D17"

Declination Source (0x0D, 0x43)

Description	Set/Get the local declination angle source. This can be used to correct for the difference in magnetic and true north. Normally, the device reports heading with-respect-to magnetic north, but when an accurate declination angle is provided, the device will report heading with respect to true north.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings *Possible declination sources:* 0x01 – None 0x02 – World Magnetic Model (Default) 0x03 – Manual Option description: None: orientation information will be reported with respect to magnetic north. World Magnetic Model: The declination will be sourced from the device's internal world magnetic model. Manual: The user provides the declination angle. The device does not validate this angle and it is therefore up to the user to select the correct value.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x08	0x43	U8 – Function Selector U8 – Declination Source Float – Manual Declination angle (radians, only required if source = Manual)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)

Reply field 2 (function = 2)	0x07		0xB2		U8 – Declination Source Float – Declination angle (radians)				
Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x08**	**0x08**	**0x43**	*Fctn (Apply):* **0x01** *Source (Manual):* **0x03** *Angle:*0x00000000 (**0.0f)**	**0x**	**0x**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x43** *Error code:***0x00**	**0x23**	**0x36**

Accelerometer Magnitude Error Adaptive Measurement (0x0D, 0x44)

Description	Set the accelerometer magnitude error adaptive measurement parameters. This function can be used to tune the filter performance in the target application.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Adaptive measurements can be enabled/disabled without the need for providing the additional parameters. In this case, only the function selector and enable value are required; all other parameters will remain at their previous values.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x1C (28)	0x44	U8 – Function Selector U8 – Enable (0 – Disable, 1 – Enable) Float – Low-pass filter cutoff frequency (Hz) Float – Low Limit (meters/second^2) Float – High Limit (meters/second^2) Float – Low Limit Uncertainty, 1-Sigma (meters/second^2) Float – High Limit Uncertainty, 1-Sigma (meters/second^2) Float – Minimum Uncertainty, 1-Sigma (meters/second^2)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x1B (27)	0xB3	U8 – Enable (0 – Disable, 1 – Enable) Float – Low-pass filter cutoff frequency (Hz) Float – Low Limit (meters/second^2) Float – High Limit (meters/second^2) Float – Low Limit Uncertainty, 1-Sigma (meters/second^2) Float – High Limit Uncertainty, 1-Sigma (meters/second^2)

			Float – Minimum Uncertainty, 1-Sigma (meters/second^2)						
Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	0x75	0x65	0x0D	0x1C	0x1C	0x44	Fctn (Apply): 0x01 Enable: 0x01 Freq (Hz):(1.0f) Low Limit:(-0.2) High Limit:(0.2f) Low Limit 1-sigma:(0.2f) High Limit 1-sigma:(0.2f) Min 1-sigma:(0.004f)	0x	0x
Reply ACK/NACK	0x75	0x65	0x0D	0x04	0x04	0xF1	Echo cmd: 0x44 Error code:0x00	0x	0x

Magnetometer Magnitude Error Adaptive Measurement (0x0D, 0x45)

Description	Set the magnetometer magnitude error adaptive measurement parameters. This function can be used to tune the filter performance in the target application.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings Adaptive measurements can be enabled/disabled without the need for providing the additional parameters. In this case, only the function selector and enable value are required; all other parameters will remain at their previous values.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x1C (28)	0x45	U8 – Function Selector U8 – Enable (0 – Disable, 1 – Enable) Float – Low-pass filter cutoff frequency (Hz) Float – Low Limit (Gauss) Float – High Limit (Gauss) Float – Low Limit Uncertainty, 1-Sigma (Gauss) Float – High Limit Uncertainty, 1-Sigma (Gauss) Float – Minimum Uncertainty, 1-Sigma (Gauss)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x1B (27)	0xB4	U8 – Enable (0 – Disable, 1 – Enable) Float – Low-pass filter cutoff frequency (Hz) Float – Low Limit (Gauss) Float – High Limit (Gauss) Float – Low Limit Uncertainty, 1-Sigma (Gauss)

				Float – High Limit Uncertainty, 1-Sigma (Gauss) Float – Minimum Uncertainty, 1-Sigma (Gauss)					
Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x1C**	**0x1C**	**0x45**	*Fctn (Apply):* **0x01** *Enable:* **0x01** *Freq (Hz):*(**1.0f)** *Low Limit:*(**-0.2)** *High Limit:*(**0.2f)** *Low Limit 1 sigma:*(**0.2f)** *High Limit 1-sigma:*(**0.2f)** *Min 1-sigma:*(**0.01f)**	**0x**	**0x**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x45** *Error code:***0x00**	**0x**	**0x**

Magnetometer Dip Angle Error Adaptive Measurement (0x0D, 0x46)

Description	Set the magnetometer dip angle error adaptive measurement parameters. This function can be used to tune the filter performance in the target application.		
Notes	*Possible function selector values:* 0x01 – Use new settings 0x02 – Read back current settings. 0x03 – Save current settings as startup settings 0x04 – Load saved startup settings 0x05 – Reset to factory default settings The adaptive function is linear, taking inputs from 0 to the stated high limit (in radians) and outputs from the minimum uncertainty to the high-limit uncertainty (in Gauss.) Adaptive measurements can be enabled/disabled without the need for providing the additional parameters. In this case, only the function selector and enable value are required; all other parameters will remain at their previous values.		
Field Format	*Field Length*	*Field Descriptor*	*Field Data*
Command	0x14 (20)	0x46	U8 – Function Selector U8 – Enable (0 – Disable, 1 – Enable) Float – Low-pass filter cutoff frequency (Hz) Float – High Limit (Radians) Float – High Limit Uncertainty, 1-Sigma (Gauss) Float – Minimum Uncertainty, 1-Sigma (Gauss)
Reply ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2	0x13 (19)	0xB5	U8 – Enable (0 – Disable, 1 – Enable) Float – Low-pass filter cutoff frequency (Hz)

(function = 2)			Float – High Limit (Radians) Float – High Limit Uncertainty, 1-Sigma (Gauss) Float – Minimum Uncertainty, 1-Sigma (Gauss)

Example	*MIP Packet Header*				*Fields*			*Checksum*	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command	**0x75**	**0x65**	**0x0D**	**0x14**	**0x14**	**0x46**	*Fctn (Apply):* **0x01** *Enable:* **0x01** *Freq (Hz):*(**10.0f)** *High Limit (rad):*(**0.3f)** *High Limit 1-sigma:*(**0.2f)** *Min 1-sigma:*(**0.01f)**	**0x**	**0x**
Reply ACK/NACK	**0x75**	**0x65**	**0x0D**	**0x04**	**0x04**	**0xF1**	*Echo cmd:* **0x46** *Error code:***0x00**	**0x**	**0x**

System Commands

The System Command set provides a set of advanced commands that are specific to devices such as the 3DM-GX3-35 that have multiple intelligent internal sensor blocks. These commands allow special mode such as talking directly to the native protocols of the embedded sensor blocks. For example, with the 3DM-GX3-35, you may switch into a mode that talks directly to the internal u-blox chip or directly to the embedded 3DM-GX3-25 IMU.

Communication Mode (0x7F, 0x10)

Advanced

<table>
<tr><td>Description</td><td colspan="3">Set, read, or save the device communication mode. This will change the communications protocol to and from “Estimation Filter” mode to “Sensor Direct” (MIP IMU protocol for the GQ4) or “GNSS Direct” (u-blox6 protocols on the GQ4). This command is always active, even when switched to the direct modes. This command responds with an ACK/NACK just prior to switching to the new protocol. For all functions except 0x01 (use new settings), the new communications mode value is ignored.</td></tr>
<tr><td>Notes</td><td colspan="3">Possible function selector values:

0x01 – Use new settings
0x02 – Read back current settings.
0x03 – Save current settings as startup settings
0x04 – Load saved startup settings
0x05 – Reset to factory default settings

Possible Communications Modes:

<table><tr><th>Value</th><th>Mode</th><th>Protocol(s)</th></tr><tr><td>0x01</td><td>Standard</td><td>3DM-GQ4-45 MIP Packet (default)</td></tr><tr><td>0x02</td><td>Sensor Direct</td><td>MIP IMU</td></tr><tr><td>0x03</td><td>GNSS Direct</td><td>NMEA, UBX</td></tr></table>
IMPORTANT: GNSS message settings are automatically when switching from direct modes back into standard mode.

Note: Switching to and from GNSS Direct Mode takes longer than most commands to complete due to the amount of GNSS setup data that needs to be stored/retrieved.</td></tr>
<tr><td>Field Format</td><td>Field</td><td>Field Descriptor</td><td>Field Data</td></tr>
</table>

	Length		
Command	0x04	0x10	U8 –Function Selector U8 –New Communications Mode
Reply field 1 ACK/NACK	0x04	0xF1	U8 – echo the command descriptor U8 – error code (0:ACK, not 0:NACK)
Reply field 2 (function = 2)	0x03	0x90	U8 –Current Communications Mode

Example	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Field Desc.*	*Field Data*	*MSB*	*LSB*
Command COM Mode	**0x75**	**0x65**	**0x7F**	**0x04**	**0x04**	**0x10**	*Fctn(USE):***0x01** New mode (*IMU Direct*): **0x02**	**0x74**	**0xBD**
Reply ACK/NACK	**0x75**	**0x65**	**0x7F**	**0x04**	**0x04**	**0xF1**	*Echo cmd:***0x10** *Error code:***0x00**	**0x62**	**0x7C**

Copy-Paste version of the command: "7565 7F04 0410 0102 74BD"

Error Codes

Error Name	*Error Value*	*Description*
MIP Unknown Command	0x01	The command descriptor is not supported by this device
MIP Invalid Checksum	0x02	An otherwise complete packet has a bad checksum
MIP Invalid Parameter	0x03	One or more parameters in the packet are invalid. This can refer to a value that is outside the allowed range for a command or a value that is not the expected size or type
MIP Command Failed	0x04	Device could not complete the command
MIP Command Timeout	0x05	Device did not complete the command within the expected time

Data Reference

IMU Data

Scaled Accelerometer Vector (0x80, 0x04)

Description	Scaled Accelerometer Vector					
Notes	This is a vector quantifying the direction and magnitude of the acceleration that the 3DM-GQ4-45 is exposed to. This quantity is fully temperature compensated and scaled into physical units of *g* (1 *g* = 9.80665 m/sec^2). It is expressed in terms of the 3DM-GQ4-45's local coordinate system.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	14 (0x0E)	0x04	0	X Accel	float	g
			4	Y Accel	float	g
			8	Z Accel	float	g

Scaled Gyro Vector (0x80, 0x05)

Description	Scaled Gyro Vector					
Notes	This is a vector quantifying the rate of rotation (angular rate) of the 3DM-GQ4-45. This quantity is fully temperature compensated and scaled into units of radians/second. It is expressed in terms of the 3DM-GQ4-45's local coordinate system.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	14 (0x0E)	0x05	0	X Gyro	float	Radians/second
			4	Y Gyro	float	Radians/second
			8	Z Gyro	float	Radians/second

Scaled Magnetometer Vector (0x80, 0x06)

Description	Scaled Magnetometer Vector					
Notes	This is a vector which gives the instantaneous magnetometer direction and magnitude. This quantity is fully temperature compensated and scaled into units of Gauss. It is expressed in terms of the 3DM-GQ4-45's local coordinate system.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	14 (0x0E)	0x06	0	X Mag	float	Gauss
			4	Y Mag	float	Gauss
			8	Z Mag	float	Gauss

Scaled Ambient Pressure (0x80, 0x17)

Description	Scaled Ambient Pressure					
Notes	This is a scalar which gives the instantaneous ambient pressure reading. This quantity is fully temperature compensated and scaled into units of milliBar.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	06 (0x06)	0x17	0	Ambient Pressure	float	milliBar

Delta Theta Vector (0x80, 0x07)

Description	Time integral of angular rate.					
Notes	This is a vector which gives the time integral of angular rate over the interval set by the IMU message format command. It is expressed in terms of the 3DM-GQ4-45's local coordinate system in units of radians.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
	14 (0x0E)	0x07	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	X Delta Theta	float	radians
			4	Y Delta Theta	float	radians
			8	Z Delta Theta	float	radians

Delta Velocity Vector (0x80, 0x08)

Description	Time integral of acceleration.					
Notes	This is a vector which gives the time integral of specific acceleration over the interval set by the IMU message format command. It is expressed in terms of the 3DM-GQ4-45's local coordinate system in units of g*second where g is the standard gravitational constant. To convert Delta Velocity into the more conventional units of m/sec, simply multiply by the standard gravitational constant, 9.80665 m/sec^2					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
	14 (0x0E)	0x08	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	X Delta Velocity	float	g*seconds
			4	Y Delta Velocity	float	g*seconds
			8	Z Delta Velocity	float	g*seconds

CF Orientation Matrix (0x80, 0x09)

Description	3 x 3 Orientation Matrix *M* *This value is produced by the Complementary Filter fusion algorithm*					
Notes	This is a 9 component coordinate transformation matrix which describes the orientation of the 3DM-GX3® with respect to the fixed earth coordinate system. $M = \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,3} \\ M_{2,1} & M_{2,2} & M_{2,3} \\ M_{3,1} & M_{3,2} & M_{3,3} \end{bmatrix}$ *M* satisfies the following equation: $V\_IL_i = M_{ij} \cdot V\_E_j$ *Where: V_IL is a vector expressed in the 3DM-GX3®'s local coordinate system.* *V_E is the same vector expressed in the stationary, earth-fixed coordinate system*					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	38 (0x26)	0x09	0	M_{11}	float	*n/a*
			4	M_{12}	float	*n/a*
			8	M_{13}	float	*n/a*
			12	M_{21}	float	*n/a*
			16	M_{22}	float	*n/a*
			20	M_{23}	float	*n/a*

			24	M_{31}	float	*n/a*
			28	M_{32}	float	*n/a*
			32	M_{33}	float	*n/a*

CF Quaternion (0x80, 0x0A)

<table>
<tr><td>Description</td><td colspan="6">4 x 1 quaternion Q.
This value is produced by the Complementary Filter fusion algorithm</td></tr>
<tr><td>Notes</td><td colspan="6">This is a 4 component quaternion which describes the orientation of the 3DM-GX3 with respect to the fixed earth coordinate quaternion.
$$Q = \begin{bmatrix} q0 \\ q1 \\ q2 \\ q3 \end{bmatrix}$$
Q satisfies the following equation:
$V\_ILi = Q^{-1} \cdot V\_E \cdot Q$
Where: V_IL is a vector expressed in the 3DM-GX3®'s local coordinate system.
V_E is the same vector expressed in the stationary, earth-fixed coordinate system</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">18 (0x12)</td><td rowspan="5">0x0A</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>q_0</td><td>float</td><td>n/a</td></tr>
<tr><td>4</td><td>q_1</td><td>float</td><td>n/a</td></tr>
<tr><td>8</td><td>q_2</td><td>float</td><td>n/a</td></tr>
<tr><td>12</td><td>q_3</td><td>float</td><td>n/a</td></tr>
</table>

CF Euler Angles (0x80, 0x0C)

<table>
<tr><td>Description</td><td colspan="6">Pitch, Roll, and Yaw (aircraft) values
This value is produced by the Complementary Filter fusion algorithm</td></tr>
<tr><td>Notes</td><td colspan="6">This is a 3 component vector containing the Roll, Pitch and Yaw angles in radians. It is computed by the IMU/AHRS from the orientation matrix M.
$$Euler = \begin{bmatrix} Roll \\ Pitch \\ Yaw \end{bmatrix}$$ (radians)</td></tr>
<tr><td rowspan="5">Field Format</td><td rowspan="2">Field Length</td><td rowspan="2">Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td rowspan="3">14 (0x0E)</td><td rowspan="3">0x0C</td><td>0</td><td>Roll</td><td>float</td><td>radians</td></tr>
<tr><td>4</td><td>Pitch</td><td>float</td><td>radians</td></tr>
<tr><td>8</td><td>Yaw</td><td>float</td><td>radians</td></tr>
</table>

CF Stabilized Mag Vector (North) (0x80, 0x10)

Description	Gyro stabilized estimated vector for geomagnetic vector. *This value is produced by the Complementary Filter fusion algorithm*					
Notes	This is a vector which represents the complementary filter's best estimate of the geomagnetic field direction (magnetic north). In the absence of magnetic interference, it should be equal to *Magnetometer*. When transient magnetic interference is present, *Magnetometer* will be subject to transient (possibly large) errors. The IMU/AHRS complementary filter computes *Stabilized North* which is its estimate of the geomagnetic field vector only, even thought the system may be exposed to transient magnetic interference. Note that sustained magnetic interference cannot be adequately compensated for by the complementary filter.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	14 (0x0E)	0x10	0	X Stab Mag	Float	Gauss
			4	Y Stab Mag	Float	Gauss
			8	Z Stab Mag	Float	Gauss

CF Stabilized Accel Vector (Up) (0x80, 0x11)

Description	Gyro stabilized estimated vector for the gravity vector. *This value is produced by the Complementary Filter fusion algorithm*					
Notes	This is a vector which represents the IMU/AHRS complementary filter's best estimate of the vertical direction. Under stationary conditions, it should be equal to Accel. In dynamic conditions, Accel will be sensitive to both gravitational acceleration as well as linear acceleration. The Complementary filter computes Stab Accel which is its estimate of the gravitation acceleration only, even thought the system may be exposed to significant linear acceleration.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	14 (0x0E)	0x11	0	X Stab Accel	Float	g
			4	Y Stab Accel	Float	g
			8	Z Stab Accel	Float	g

GPS Correlation Timestamp (0x80, 0x12)

<table>
<tr><td>Description</td><td colspan="6">GPS correlation timestamp.</td></tr>
<tr><td>Notes</td><td colspan="6">This timestamp has three fields:

Double GPS TOW
U16 GPS Week number
U16 Timestamp flags

Timestamp Status Flags:

Bit0 – PPS Beacon Good If set, GNSS receiver PPS signal is present
Bit1 – GPS Time Refresh (toggles with each refresh)
Bit2 – GPS Time Initialized (set with the first GPS Time Refresh)

This record (GPS Correlation Timestamp) is identical to a GPS Time record from the GNSS receiver except the flags are defined specifically for the IMU. When the GPS Time Initialized flag is asserted, the GPS Time and IMU GPS Timestamp are correlated. This flag is only set once upon the first valid GPS Time record. After that, each time the GPS Time becomes invalid (from a lack of signal) and then valid again (regains signal) the GPS Time Refresh flag will toggle. The GPS Time Initialized will remain set.

The “PPS Beacon Good” flag in the Timestamp flags byte indicates if the PPS beacon coming from the GNSS receiver is present. If this flag is not asserted, it means that the IMU internal clock is being used for the PPS. The fractional portion of the GPS TOW represents the amount of time that has elapsed from the last PPS.

If the GNSS receiver loses signal, the GNSS receiver and IMU timestamps become free running and will slowly drift away from each other. If the timestamp clocks have drifted apart, then there will be a jump in the timestamp when the PPS Beacon Good reasserts, reflecting the amount of drift of the clocks.

See the Data Synchronicity section of this manual for more information on timestamps.

⚠ It is important to understand that new GNSS constellations such as GLONASS and BeiDou do not use the GPS Time base (the time base of the U.S. GPS constellation). However, their time bases get converted to the GPS Time base by the receiver.</td></tr>
<tr><td rowspan="4">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="3">14 (0x0E)</td><td rowspan="3">0x12</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>GPS Time of Week</td><td>Double</td><td>Seconds</td></tr>
<tr><td>8</td><td>GPS Week</td><td>U16</td><td></td></tr>
</table>

				Number		
			10	Timestamp Flags	U16	See Notes

GNSS Data

LLH Position (0x81, 0x03)

Description	Position Data in the Geodetic Frame					
Notes	Valid Flag Mapping: 0x0001 – Latitude & Longitude Valid 0x0002 – Ellipsoid Height Valid 0x0004 – MSL Height Valid 0x0008 – Horizontal Accuracy Valid 0x0010 – Vertical Accuracy Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	44 (0x2C)	0x03	0	Latitude	Double	Decimal Degrees
			8	Longitude	Double	Decimal Degrees
			16	Height above Ellipsoid	Double	Meters
			24	Height above MSL	Double	Meters
			32	Horizontal Accuracy	Float	Meters
			36	Vertical Accuracy	Float	Meters
			40	Valid Flags	U16	See Notes

ECEF Position (0x81, 0x04)

<table>
<tr><td>Description</td><td colspan="6">Position Data in the Earth-Centered, Earth-Fixed Frame</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0001 – ECEF Position Valid
0x0002 – Position Accuracy Valid</td></tr>
<tr><td rowspan="7">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="6">32 (0x20)</td><td rowspan="6">0x04</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X Position</td><td>Double</td><td>Meters</td></tr>
<tr><td>8</td><td>Y Position</td><td>Double</td><td>Meters</td></tr>
<tr><td>16</td><td>Z Position</td><td>Double</td><td>Meters</td></tr>
<tr><td>24</td><td>Position Accuracy</td><td>Float</td><td>Meters</td></tr>
<tr><td>28</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

NED Velocity (0x81, 0x05)

<table>
<tr><td>Description</td><td colspan="6">Velocity Data in the North-East-Down Frame</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0001 – NED Velocity Valid
0x0002 – Speed Valid
0x0004 – Ground Speed Valid
0x0008 – Heading Valid
0x0010 – Speed Accuracy Valid
0x0020 – Heading Accuracy Valid</td></tr>
<tr><td rowspan="11">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="10">36(0x24)</td><td rowspan="10">0x05</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>North</td><td>Float</td><td>Meters / Sec</td></tr>
<tr><td>4</td><td>East</td><td>Float</td><td>Meters / Sec</td></tr>
<tr><td>8</td><td>Down</td><td>Float</td><td>Meters / Sec</td></tr>
<tr><td>12</td><td>Speed</td><td>Float</td><td>Meters / Sec</td></tr>
<tr><td>16</td><td>Ground Speed</td><td>Float</td><td>Meters / Sec</td></tr>
<tr><td>20</td><td>Heading</td><td>Float</td><td>Decimal Degrees</td></tr>
<tr><td>24</td><td>Speed Accuracy</td><td>Float</td><td>Meters / Sec</td></tr>
<tr><td>28</td><td>Heading Accuracy</td><td>Float</td><td>Decimal Degrees</td></tr>
<tr><td>32</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

ECEF Velocity (0x81, 0x06)

Description	Velocity Data in the Earth-Centered, Earth-Fixed Frame
Notes	Valid Flag Mapping: 0x0001 – ECEF Velocity Valid 0x0002 – Velocity Accuracy Valid

	Field Length	Data Descriptor	Message Data			
			Binary Offset	*Description*	*Data Type*	*Units*
Field Format	20 (0x14)	0x06	0	X Velocity	Float	Meters / Sec
			4	Y Velocity	Float	Meters / Sec
			8	Z Velocity	Float	Meters / Sec
			12	Velocity Accuracy	Float	Meters / Sec
			16	Valid Flags	U16	See Notes

DOP Data (0x81, 0x07)

<table>
<tr><td>Description</td><td colspan="6">Dilution of Precision Data</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0001 – GDOP Valid
0x0002 – PDOP Valid
0x0004 – HDOP Valid
0x0008 – VDOP Valid
0x0010 – TDOP Valid
0x0020 – NDOP Valid
0x0040 – EDOP Valid</td></tr>
<tr><td rowspan="10">Field Format</td><td rowspan="2">Field Length</td><td rowspan="2">Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td rowspan="8">32 (0x20)</td><td rowspan="8">0x07</td><td>0</td><td>Geometric DOP</td><td>Float</td><td>N/A</td></tr>
<tr><td>4</td><td>Position DOP</td><td>Float</td><td>N/A</td></tr>
<tr><td>8</td><td>Horizontal DOP</td><td>Float</td><td>N/A</td></tr>
<tr><td>12</td><td>Vertical DOP</td><td>Float</td><td>N/A</td></tr>
<tr><td>16</td><td>Time DOP</td><td>Float</td><td>N/A</td></tr>
<tr><td>20</td><td>Northing DOP</td><td>Float</td><td>N/A</td></tr>
<tr><td>24</td><td>Easting DOP</td><td>Float</td><td>N/A</td></tr>
<tr><td>28</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

UTC Time (0x81, 0x08)

Description	Coordinated Universal Time Data					
Notes	Valid Flag Mapping: 0x0001 – Date Valid 0x0002 – Time Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	15 (0x0F)	0x08	0	Year	U16	Years (1999-2099)
			2	Month	U8	Months (1-12)
			3	Day	U8	Days (1-31)
			4	Hour	U8	Hours (0-23)
			5	Minute	U8	Minutes (0-59)
			6	Second	U8	Seconds (0-59)
			7	Millisecond	U32	Milliseconds
			11	Valid Flags	U16	See Notes

GPS Time (0x81, 0x09)

Description	Global Positioning System Time Data					
Notes	Valid Flag Mapping: 0x0001 – TOW Valid 0x0002 – Week Number Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	14 (0x0E)	0x09	0	Time of Week	Double	Seconds
			8	Week Number	U16	
			10	Valid Flags	U16	See Notes

Clock Information (0x81, 0x0A)

<table>
<tr><td>Description</td><td colspan="6">Detailed information about the GNSS Clock</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0001 – Bias Valid
0x0002 – Drift Valid
0x0004 – Accuracy Estimate Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">28(0x1C)</td><td rowspan="5">0x0A</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>Clock Bias</td><td>Double</td><td>Seconds</td></tr>
<tr><td>8</td><td>Clock Drift</td><td>Double</td><td>Seconds/Second</td></tr>
<tr><td>16</td><td>Accuracy Estimate</td><td>Double</td><td>Seconds</td></tr>
<tr><td>24</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

GNSS Fix Information (0x81, 0x0B)

<table>
<tr><td>Description</td><td colspan="6">Current GNSS Fix Status Information</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0001 – Fix Type Valid
0x0002 – Number of SVs Valid
0x0004 – Fix Flags Valid

Possible Fix Types values are:

0x00 – 3D Fix
0x01 – 2D Fix
0x02 – Time Only
0x03 – None
0x04 – Invalid

Possible Fix Flags are:

0x0001 – SBAS Corrections Used
0x0002 – Differential (DGNSS) Corrections Used</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">8(0x08)</td><td rowspan="5">0x0B</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>Fix Type</td><td>U8</td><td>See Notes</td></tr>
<tr><td>1</td><td>Number of SVs used for solution</td><td>U8</td><td>Count</td></tr>
<tr><td>2</td><td>Fix Flags (Reserved)</td><td>U16</td><td>N/A</td></tr>
<tr><td>4</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Space Vehicle Information (0x81, 0x0C)

<table>
<tr><td>Description</td><td colspan="6">Individual Space Vehicle Information Entry</td></tr>
<tr><td>Notes</td><td colspan="6">When enabled, these fields will arrive in a separate MIP packet.

Valid Flag Mapping:

0x0001 – Channel Valid
0x0002 – SV ID Valid
0x0008 – Carrier to Noise Ratio Valid
0x0010 – Azimuth Valid
0x0020 – Elevation Valid
0x0040 – SV Flags Valid

SV Flag Mapping:

0x0001 – SV Used for Navigation
0x0002 – SV Healthy</td></tr>
<tr><td rowspan="9">Field Format</td><td rowspan="2">Field Length</td><td rowspan="2">Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td rowspan="7">14(0x0E)</td><td rowspan="7">0x0C</td><td>0</td><td>Channel</td><td>U8</td><td>Channel Number</td></tr>
<tr><td>1</td><td>Space Vehicle ID</td><td>U8</td><td>SV ID Number</td></tr>
<tr><td>2</td><td>Carrier to Noise Ratio</td><td>U16</td><td>dBHz</td></tr>
<tr><td>4</td><td>Azimuth</td><td>S16</td><td>Integer Degrees</td></tr>
<tr><td>6</td><td>Elevation</td><td>S16</td><td>Integer Degrees</td></tr>
<tr><td>8</td><td>Space Vehicle Flags</td><td>U16</td><td>See Notes</td></tr>
<tr><td>10</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Hardware Status (0x81, 0x0D)

<table>
<tr><td>Description</td><td colspan="6">GNSS Hardware Status Information</td></tr>
<tr><td>Notes</td><td colspan="6">Hardware status is only available at 1 Hz. Setting the rate higher than 1 Hz has no effect.

Valid Flag Mapping:

0x0001 – Sensor State Valid
0x0002 – Antenna State Valid
0x0004 – Antenna Power Valid

Possible Sensor State values:

0x00 – Sensor Off
0x01 – Sensor On
0x02 – Sensor State Unknown

Possible Antenna State values:

0x01 – Antenna Init
0x02 – Antenna Short
0x03 – Antenna Open
0x04 – Antenna Good
0x05 – Antenna State Unknown.

Possible Antenna Power values:

0x00 – Antenna Off
0x01 – Antenna On
0x02 – Antenna Power Unknown</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">7(0x07)</td><td rowspan="5">0x0D</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>Sensor State</td><td>U8</td><td>See Notes</td></tr>
<tr><td>1</td><td>Antenna State</td><td>U8</td><td>See Notes</td></tr>
<tr><td>2</td><td>Antenna Power</td><td>U8</td><td>See Notes</td></tr>
<tr><td>3</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

DGNSS Information (0x81, 0x0E)

<table>
<tr><td>Description</td><td colspan="6">Individual DGNSS Channel Status Entry</td></tr>
<tr><td>Notes</td><td colspan="6">When enabled, a separate field for each active space vehicle will be sent in the packet.

Valid Flag Mapping:

0x0001 – Latest Age Valid
0x0002 – Base Station ID Valid
0x0004 – Base Station Status Valid
0x0008 – Number of DGNSS Channels Valid

Possible Base Station Status Values:

0 – UDRE Scale Factor = 1.0
1 – UDRE Scale Factor = 0.75
2 – UDRE Scale Factor = 0.5
3 – UDRE Scale Factor = 0.3
4 – UDRE Scale Factor = 0.2
5 – UDRE Scale Factor = 0.1
6 – Reference Station Transmission Not Monitored
7 – Reference Station Not Working

Note: UDRE = User Differential Range Error</td></tr>
<tr><td rowspan="7">Field Format</td><td rowspan="2">Field Length</td><td rowspan="2">Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td rowspan="5">14 (0x0E)</td><td rowspan="5">0x0E</td><td>0</td><td>Newest Age</td><td>float</td><td>Seconds</td></tr>
<tr><td>4</td><td>Base Station ID</td><td>S16</td><td>N/A</td></tr>
<tr><td>6</td><td>Base Station Status</td><td>S16</td><td>N/A</td></tr>
<tr><td>8</td><td>Number of DGNSS Channels</td><td>U16</td><td>Number</td></tr>
<tr><td>10</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

DGNSS Channel Status (0x81, 0x0F)

<table>
<tr><td>Description</td><td colspan="6">Individual DGNSS Channel Status Entry</td></tr>
<tr><td>Notes</td><td colspan="6">When enabled, a separate field for each active space vehicle will be sent in the packet.

Valid Flag Mapping:

0x0001 – SV ID Valid
0x0002 – Age Valid
0x0004 – Pseudorange Correction Valid
0x0008 – Pseudorange Rate Correction Valid</td></tr>
<tr><td rowspan="7">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="6">17(0x11)</td><td rowspan="6">0x0F</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>Space Vehicle ID</td><td>U8</td><td>SV ID Number</td></tr>
<tr><td>1</td><td>Age</td><td>float</td><td>Seconds</td></tr>
<tr><td>5</td><td>Pseudorange Correction</td><td>float</td><td>Meters</td></tr>
<tr><td>9</td><td>Pseudorange Rate Correction</td><td>float</td><td>Meters/Sec</td></tr>
<tr><td>13</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Estimation Filter Data

Filter Status (0x82, 0x10)

Description	Estimation Filter Status
Notes	Possible Filter States: 0x00 – Startup 0x01 – Initialization (see status flags) 0x02 – Running, Solution Valid 0x03 – Running, Solution Error (see status flags) Possible Dynamics Modes: 0x01 – Portable 0x02 – Automotive 0x03 – Airborne Possible Status Flags: Filter State = Initialization: 0x1000 – Attitude not initialized 0x2000 – Position & Velocity not initialized Filter State = Running: 0x0001 – IMU Unavailable 0x0002 – GNSS Unavailable 0x0008 – Matrix Singularity in calculation 0x0010 – Position Covariance High Warning* 0x0020 – Velocity Covariance High Warning* 0x0040 – Attitude Covariance High Warning* 0x0080 – NAN in Solution 0x0100 – Gyro bias estimate high warning 0x0200 – Accel bias estimate high warning 0x0400 – Gyro scale factor estimate high warning 0x0800 – Accel scale factor estimate high warning 0x2000 – GNSS Antenna Offset Correction estimate high warning *Note: The covariance high warnings are triggered when **any** axis of the

	covariance vector exceeds normal operating limits. If more information is required, please inspect the relevant uncertainty packet to determine which axis is in error.					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	08 (0x08)	0x10	0	Filter State	U16	See Notes
			2	Dynamics Mode	U16	See Notes
			4	Status Flags	U16	See Notes

GPS Timestamp (0x82, 0x11)

Description	Estimation Filter Calculated Value Timestamp Data					
Notes	Valid Flag Mapping: 0x0000 – Time Invalid 0x0001 – Time Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	14 (0x0E)	0x11	0	Time of Week	Double	Seconds
			8	Week Number	U16	
			10	Valid Flags	U16	See Notes

LLH Position (0x82, 0x01)

Description	Estimated Position Data expressed in the Geodetic Frame					
Notes	Valid Flag Mapping: 0x0000 – Latitude, Longitude, & Height are Invalid 0x0001 – Latitude, Longitude, & Height Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	28 (0x1C)	0x01	0	Latitude	Double	Decimal Degrees
			8	Longitude	Double	Decimal Degrees
			16	Height above Ellipsoid	Double	Meters
			24	Valid Flags	U16	See Notes

NED Velocity (0x82, 0x02)

Description	Estimated Velocity Data expressed in the Local-Level Frame					
Notes	Valid Flag Mapping: 0x0000 – NED Velocity is Invalid 0x0001 – NED Velocity Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	16 (0x10)	0x02	0	North	Float	Meters / Sec
			4	East	Float	Meters / Sec
			8	Down	Float	Meters / Sec
			12	Valid Flags	U16	See Notes

Orientation, Quaternion (0x82, 0x03)

<table>
<tr><td>Description</td><td colspan="6">Estimated Orientation in quaternion form.</td></tr>
<tr><td>Notes</td><td colspan="6">This is a 4 component quaternion which describes the orientation of the 3DM-GQ4-45 with respect to the fixed earth coordinate quaternion.
$$Q = \begin{bmatrix} q0 \\ q1 \\ q2 \\ q3 \end{bmatrix}$$
Q satisfies the following equation:
V_E = Q • V_IL • Q-1
Where: V_IL is a vector expressed in the 3DM-GQ4®'s local coordinate system.
V_E is the same vector expressed in the stationary, earth-fixed coordinate system
Valid Flag Mapping:
0x0000 – Quaternion is Invalid
0x0001 – Quaternion Valid</td></tr>
<tr><td rowspan="7">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="6">20 (0x14)</td><td rowspan="6">0x03</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>q_0</td><td>float</td><td>n/a</td></tr>
<tr><td>4</td><td>q_1* i</td><td>float</td><td>n/a</td></tr>
<tr><td>8</td><td>q_2* j</td><td>float</td><td>n/a</td></tr>
<tr><td>12</td><td>q_3* k</td><td>float</td><td>n/a</td></tr>
<tr><td>16</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Orientation, Matrix (0x82, 0x04)

<table>
<tr><td>Description</td><td colspan="6">Estimated Orientation in Matrix form.</td></tr>
<tr><td>Notes</td><td colspan="6">This is a 9 component coordinate transformation matrix which describes the orientation of the 3DM-GQ4®with respect to the fixed earth coordinate system.

$$M = \begin{bmatrix} M_{1,1} & M_{1,2} & M_{1,3} \\ M_{2,1} & M_{2,2} & M_{2,3} \\ M_{3,1} & M_{3,2} & M_{3,3} \end{bmatrix}$$

M satisfies the following equation:

$V\_IL_i = M_{ij} \cdot V\_E_j$

Where: V_IL is a vector expressed in the 3DM-GQ4®'s local coordinate system.
V_E is the same vector expressed in the stationary, earth-fixed coordinate system

Valid Flag Mapping:

0x0000 – Orientation Matrix is Invalid
0x0001 – Orientation Matrix Valid</td></tr>
<tr><td rowspan="11">Field Format</td><td rowspan="2">Field Length</td><td rowspan="2">Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td rowspan="9">40 (0x28)</td><td rowspan="9">0x04</td><td>0</td><td>M_{11}</td><td>float</td><td>n/a</td></tr>
<tr><td>4</td><td>M_{12}</td><td>float</td><td>n/a</td></tr>
<tr><td>8</td><td>M_{13}</td><td>float</td><td>n/a</td></tr>
<tr><td>12</td><td>M_{21}</td><td>float</td><td>n/a</td></tr>
<tr><td>16</td><td>M_{22}</td><td>float</td><td>n/a</td></tr>
<tr><td>20</td><td>M_{23}</td><td>float</td><td>n/a</td></tr>
<tr><td>24</td><td>M_{31}</td><td>float</td><td>n/a</td></tr>
<tr><td>28</td><td>M_{32}</td><td>float</td><td>n/a</td></tr>
<tr><td>32</td><td>M_{33}</td><td>float</td><td>n/a</td></tr>
</table>

			36	Valid Flags	U16	See Notes

Orientation, Euler Angles (0x82, 0x05)

Description	Estimated Pitch, Roll, and Yaw (aircraft) angles					
Notes	This is a 3 component vector containing the Roll, Pitch and Yaw angles in radians. It is computed by the INS from the orientation quaternion Q. $Euler = \begin{bmatrix} Roll \\ Pitch \\ Yaw \end{bmatrix}$ (radians) Valid Flag Mapping: 0x0000 – Euler Angles are Invalid 0x0001 – Euler Angles Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	16 (0x10)	0x05	0	Roll	float	radians
			4	Pitch	float	radians
			8	Yaw	float	radians
			12	Valid Flags	U16	See Notes

Gyro Bias (0x82, 0x06)

<table>
<tr><td>Description</td><td colspan="6">Estimated Gyro Biases expressed in the Sensor Body Frame.</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0000 – Gyro Bias are Invalid
0x0001 – Gyro Bias Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">16 (0x10)</td><td rowspan="5">0x06</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X Gyro Bias</td><td>float</td><td>radians/sec</td></tr>
<tr><td>4</td><td>Y Gyro Bias</td><td>float</td><td>radians/sec</td></tr>
<tr><td>8</td><td>Z Gyro Bias</td><td>float</td><td>radians/sec</td></tr>
<tr><td>12</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Accel Bias (0x82, 0x07)

<table>
<tr><td>Description</td><td colspan="6">Estimated Accel Biases expressed in the Sensor Body Frame.</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0000 – Accel Bias are Invalid
0x0001 – Accel Bias Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">16 (0x10)</td><td rowspan="5">0x07</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X Accel Bias</td><td>float</td><td>Meters / Sec^2</td></tr>
<tr><td>4</td><td>Y Accel Bias</td><td>float</td><td>Meters / Sec^2</td></tr>
<tr><td>8</td><td>Z Accel Bias</td><td>float</td><td>Meters / Sec^2</td></tr>
<tr><td>12</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

LLH Position Uncertainty (0x82, 0x08)

Description	Estimated Position 1-sigma Uncertainty expressed in the Geodetic Frame					
Notes	Valid Flag Mapping: 0x0000 – Position Uncertainties are Invalid 0x0001 – Position Uncertainties Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
	16 (0x10)	0x08	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	Local-Level, 1-Sigma Position Uncertainty (North)	Float	Meters
			4	Local-Level, 1-Sigma Position Uncertainty (East)	Float	Meters
			8	Local-Level, 1-Sigma Position Uncertainty (Down)	Float	Meters
			12	Valid Flags	U16	See Notes

NED Velocity Uncertainty (0x82, 0x09)

Description	Estimated Velocity 1-sigma Uncertainty expressed in the Local-Level Frame					
Notes	Valid Flag Mapping: 0x0000 – NED Velocity Uncertainties are Invalid 0x0001 – NED Velocity Uncertainties Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
	16 (0x10)	0x09	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	Local-Level, 1-Sigma Velocity Uncertainty	Float	Meters / Sec

				(North)		
			4	Local-Level, 1-Sigma Velocity Uncertainty (East)	Float	Meters / Sec
			8	Local-Level, 1-Sigma Velocity Uncertainty (Down)	Float	Meters / Sec
			12	Valid Flags	U16	See Notes

Attitude Uncertainty, Euler Angles (0x82, 0x0A)

Description	Estimated attitude 1-sigma uncertainty expressed in Pitch, Roll, and Yaw (aircraft) elements.					
Notes	This is a 3 component vector containing the Roll, Pitch and Yaw angle uncertainties in radians. **IMPORTANT:** These values are derived from the quaternion elements and become increasingly inaccurate as the pitch angle approaches +-90 degrees. To compensate for this limitation, these values will be marked as invalid when the pitch angle exceeds +-70 degrees. Valid Flag Mapping: 0x0000 – Attitude Uncertainties are Invalid 0x0001 – Attitude Uncertainties Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	16 (0x10)	0x0A	0	1-Sigma Attitude Uncertainty (Roll)	float	radians
			4	1-Sigma Attitude Uncertainty	float	radians

				(Pitch)		
			8	1-Sigma Attitude Uncertainty (Yaw)	float	radians
			12	Valid Flags	U16	See Notes

Gyro Bias Uncertainty (0x82, 0x0B)

Description	Estimated Gyro Bias 1-sigma Uncertainty expressed in the Sensor Body Frame.					
Notes	Valid Flag Mapping: 0x0000 – Gyro Bias Uncertainties are Invalid 0x0001 – Gyro Bias Uncertainties Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	16 (0x10)	0x0B	0	1-Sigma Gyro Bias Uncertainty (X)	float	radians/sec
			4	1-Sigma Gyro Bias Uncertainty (Y)	float	radians/sec
			8	1-Sigma Gyro Bias Uncertainty (Z)	float	radians/sec
			12	Valid Flags	U16	See Notes

Accel Bias Uncertainty (0x82, 0x0C)

<table>
<tr><td>Description</td><td colspan="6">Estimated Accel Bias 1-sigma Uncertainty expressed in the Sensor Body Frame.</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0000 – Accel Bias Uncertainties are Invalid
0x0001 – Accel Bias Uncertainties Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">16 (0x10)</td><td rowspan="5">0x0C</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>1-Sigma Accel Bias Uncertainty (X)</td><td>float</td><td>Meters / Sec^2</td></tr>
<tr><td>4</td><td>1-Sigma Accel Bias Uncertainty (Y)</td><td>float</td><td>Meters / Sec^2</td></tr>
<tr><td>8</td><td>1-Sigma Accel Bias Uncertainty (Z)</td><td>float</td><td>Meters / Sec^2</td></tr>
<tr><td>12</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Linear Acceleration (0x82, 0x0D)

<table>
<tr><td>Description</td><td colspan="6">Filter-Compensated Linear Acceleration Data (gravity vector removed) expressed in:
1) The Sensor Frame, if no sensor to body rotation has been defined.
2) The Vehicle Frame, if a sensor to body rotation has been defined.</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0000 – Linear Accelerations are Invalid
0x0001 – Linear Accelerations are Valid</td></tr>
<tr><td rowspan="3">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="2">16 (0x10)</td><td rowspan="2">0x0D</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X</td><td>Float</td><td>Meters / Sec^2</td></tr>
</table>

			4	Y	Float	Meters / Sec^2
			8	Z	Float	Meters / Sec^2
			12	Valid Flags	U16	See Notes

Compensated Acceleration (0x82, 0x1C)

Description	Filter-Compensated Acceleration Data expressed in: 1) The Sensor Frame, if no sensor to body rotation has been defined. 2) The Vehicle Frame, if a sensor to body rotation has been defined.					
Notes	Valid Flag Mapping: 0x0000 – Compensated Accelerations are Invalid 0x0001 – Compensated Accelerations are Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	16 (0x10)	0x0D	0	X	Float	Meters / Sec^2
			4	Y	Float	Meters / Sec^2
			8	Z	Float	Meters / Sec^2
			12	Valid Flags	U16	See Notes

Compensated Angular Rate (0x82, 0x0E)

<table>
<tr><td>Description</td><td colspan="6">Filter-Compensated Angular Rate Data expressed in:
1) The Sensor Frame, if no sensor to body rotation has been defined.
2) The Vehicle Frame, if a sensor to body rotation has been defined.</td></tr>
<tr><td>Notes</td><td colspan="6">The estimated gyro bias has been removed from these angular rate values.

Valid Flag Mapping:

0x0000 – Angular Rates are not Valid
0x0001 – Angular Rates are Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">16 (0x10)</td><td rowspan="5">0x0E</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X</td><td>Float</td><td>Radians / Sec</td></tr>
<tr><td>4</td><td>Y</td><td>Float</td><td>Radians / Sec</td></tr>
<tr><td>8</td><td>Z</td><td>Float</td><td>Radians / Sec</td></tr>
<tr><td>12</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

WGS84 Local Gravity Magnitude (0x82, 0x0F)

<table>
<tr><td>Description</td><td colspan="6">Local Magnitude of Earth’s gravity using the WGS84 gravity model.</td></tr>
<tr><td>Notes</td><td colspan="6">The GQ4-45 implements the WGS84 gravity model, valid for altitudes of 20km or less.

Valid Flag Mapping:

0x0000 – Gravity value is Invalid
0x0001 – Gravity value is Valid</td></tr>
<tr><td rowspan="4">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="3">08(0x08)</td><td rowspan="3">0x0F</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>Gravity Magnitude</td><td>Float</td><td>meters / sec^2</td></tr>
<tr><td>4</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Attitude Uncertainty, Quaternion Elements (0x82, 0x12)

<table>
<tr><td>Description</td><td colspan="6">Estimated attitude 1-sigma uncertainty expressed in quaternion components.</td></tr>
<tr><td>Notes</td><td colspan="6">This is a 4 component vector containing the attitude uncertainty expressed in quaternion elements.

Valid Flag Mapping:

0x0000 – Attitude uncertainties are Invalid
0x0001 – Attitude uncertainties are Valid</td></tr>
<tr><td rowspan="7">Field Format</td><td rowspan="2">Field Length</td><td rowspan="2">Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td rowspan="5">18 (0x12)</td><td rowspan="5">0x12</td><td>0</td><td>1-Sigma Attitude Uncertainty (q0)</td><td>float</td><td></td></tr>
<tr><td>4</td><td>1-Sigma Attitude Uncertainty (q1)</td><td>float</td><td></td></tr>
<tr><td>8</td><td>1-Sigma Attitude Uncertainty (q2)</td><td>float</td><td></td></tr>
<tr><td>12</td><td>1-Sigma Attitude Uncertainty (q3)</td><td>float</td><td></td></tr>
<tr><td>16</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Gravity Vector (0x82, 0x13)

<table>
<tr><td>Description</td><td colspan="6">Estimated Gravity Vector expressed in:
1) The Sensor Frame, if no sensor to body rotation has been defined.
2) The Vehicle Frame, if a sensor to body rotation has been defined.</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0000 – Gravity vector is Invalid
0x0001 – Gravity vector is Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">16 (0x10)</td><td rowspan="5">0x13</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X</td><td>Float</td><td>Meters / Sec^2</td></tr>
<tr><td>4</td><td>Y</td><td>Float</td><td>Meters / Sec^2</td></tr>
<tr><td>8</td><td>Z</td><td>Float</td><td>Meters / Sec^2</td></tr>
<tr><td>12</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Heading Update Source State (0x82, 0x14)

Description	Heading Update Source information expressed in the sensor frame.
Notes	Heading updates can be applied from a number of sources (listed below.) The heading value is always relative to true north. Possible Sources: 0x0000 – No source, heading updates disabled 0x0001 – Internal Magnetometer 0x0002 – Internal GNSS Velocity Vector 0x0004 – External Heading Update Command Valid Flag Mapping: 0x0000 – No heading update received in 2 seconds. 0x0001 – The heading update source has provided data within 2 seconds.

	Field Length	Data Descriptor	Message Data			
			Binary Offset	Description	Data Type	Units
Field Format	14(0x0E)	0x14	0	Heading (True)	Float	Radians
			4	Heading 1-sigma Uncertainty	Float	Radians
			8	Source	U16	See Notes
			10	Valid Flags	U16	See Notes

Magnetic Model Solution (0x82, 0x15)

Description	Magnetic model solution expressed in the NED frame.
Notes	The World Magnetic Model 2010 is used. A valid GNSS location is required for the model to be valid. Valid Flag Mapping: 0x0000 – Magnetic model solution is invalid (note: this will be the state when the magnetic model is recalculating for the current time and location as well as when GNSS is unavailable) 0x0001 – Magnetic model solution is valid

	Field Length	Data Descriptor	Message Data			
			Binary Offset	Description	Data Type	Units
Field Format	24 (0x18)	0x15	0	Intensity (North)	Float	Gauss
			4	Intensity (East)	Float	Gauss
			8	Intensity (Down)	Float	Gauss
			12	Inclination	Float	Radians
			16	Declination	Float	Radians
			20	Valid Flags	U16	See Notes

Gyro Scale Factor (0x82, 0x16)

<table>
<tr><td>Description</td><td colspan="6">Estimated Gyro Scale Factor expressed in the Sensor Body Frame.</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0000 – Scale Factor values are Invalid
0x0001 – Scale Factor values are Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">16 (0x10)</td><td rowspan="5">0x16</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X Scale Factor</td><td>float</td><td>%/100</td></tr>
<tr><td>4</td><td>Y Scale Factor</td><td>float</td><td>%/100</td></tr>
<tr><td>8</td><td>Z Scale Factor</td><td>float</td><td>%/100</td></tr>
<tr><td>12</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Accel Scale Factor (0x82, 0x17)

<table>
<tr><td>Description</td><td colspan="6">Estimated Accel Scale Factor expressed in the Sensor Body Frame.</td></tr>
<tr><td>Notes</td><td colspan="6">Valid Flag Mapping:

0x0000 – Scale Factor values are Invalid
0x0001 – Scale Factor values are Valid</td></tr>
<tr><td rowspan="6">Field Format</td><td>Field Length</td><td>Data Descriptor</td><td colspan="4">Message Data</td></tr>
<tr><td rowspan="5">16 (0x10)</td><td rowspan="5">0x17</td><td>Binary Offset</td><td>Description</td><td>Data Type</td><td>Units</td></tr>
<tr><td>0</td><td>X Scale Factor</td><td>float</td><td>%/100</td></tr>
<tr><td>4</td><td>Y Scale Factor</td><td>float</td><td>%/100</td></tr>
<tr><td>8</td><td>Z Scale Factor</td><td>float</td><td>%/100</td></tr>
<tr><td>12</td><td>Valid Flags</td><td>U16</td><td>See Notes</td></tr>
</table>

Gyro Scale Factor Uncertainty (0x82, 0x18)

Description	Estimated Gyro Scale Factor 1-Sigma Uncertainty expressed in the Sensor Body Frame.					
Notes	Valid Flag Mapping: 0x0000 – Gyro Scale Factor Uncertainties are Invalid 0x0001 – Gyro Scale Factor Uncertainties Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	16 (0x10)	0x18	0	1-Sigma Gyro Scale Factor Uncertainty (X)	float	%/100
			4	1-Sigma Gyro Scale Factor Uncertainty (Y)	float	%/100
			8	1-Sigma Gyro Scale Factor Uncertainty (Z)	float	%/100
			12	Valid Flags	U16	See Notes

Accel Scale Factor Uncertainty (0x82, 0x19)

Description	Estimated Accel Scale Factor 1-Sigma Uncertainty expressed in the Sensor Body Frame.					
Notes	Valid Flag Mapping: 0x0000 – Accel Scale Factor Uncertainties are Invalid 0x0001 – Accel Scale Factor Uncertainties Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	16 (0x10)	0x19	0	1-Sigma Accel Scale Factor Uncertainty (X)	float	%/100
			4	1-Sigma Accel Scale Factor Uncertainty (Y)	float	%/100
			8	1-Sigma Accel Scale Factor Uncertainty (Z)	float	%/100
			12	Valid Flags	U16	See Notes

Standard Atmosphere Model (0x82, 0x20)

Description	Standard Atmosphere Model Solution.		
Notes	The US 1976 Standard Atmosphere Model is used. A valid GNSS location is required for the model to be valid. Valid Flag Mapping: 0x0000 – Atmosphere model solution is invalid (note: this will be the state when GNSS is unavailable) 0x0001 – Atmosphere model solution is valid		
Field Format	*Field Length*	*Data Descriptor*	*Message Data*

			Binary Offset	Description	Data Type	Units
	24 (0x18)	0x20	0	Geometric Altitude	Float	meters
			4	Geopotential Altitude	Float	meters
			8	Temperature	Float	degC
			12	Pressure	Float	mBar
			16	Density	Float	kg/m^3
			20	Valid Flags	U16	See Notes

Pressure Altitude (0x82, 0x21)

Description	Estimated Pressure Altitude.
Notes	The US 1976 Standard Atmosphere Model is used to calculate the pressure altitude in meters. A valid pressure sensor reading is required for the pressure altitude to be valid. The minimum pressure reading supported by the model is 0.0037 mBar, corresponding to an altitude of 84,852 meters. Valid Flag Mapping: 0x0000 – Pressure Altitude is Invalid 0x0001 – Pressure Altitude is Valid

Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
			Binary Offset	*Description*	*Data Type*	*Units*
	8 (0x08)	0x21	0	Pressure Altitude	float	meters
			4	Valid Flags	U16	See Notes

GNSS Antenna Offset Correction (0x82, 0x30)

Description	Estimated GNSS Antenna Offset Correction expressed in the Sensor Body Frame as the vector from the IMU to the GNSS Antenna.					
Notes	Valid Flag Mapping: 0x0000 – GNSS Antenna Offset Correction values are Invalid 0x0001 – GNSS Antenna Offset Correction values are Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
	16 (0x10)	0x30	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	X Offset Correction	float	meters
			4	Y Offset Correction	float	meters
			8	Z Offset Correction	float	meters
			12	Valid Flags	U16	See Notes

GNSS Antenna Offset Correction Uncertainty (0x82, 0x31)

Description	Estimated GNSS Antenna Offset Correction Uncertainty expressed in the Sensor Body Frame.					
Notes	Valid Flag Mapping: 0x0000 – GNSS Antenna Offset Correction Uncertainties are Invalid 0x0001 – GNSS Antenna Offset Correction Uncertainties Valid					
Field Format	*Field Length*	*Data Descriptor*	*Message Data*			
	16 (0x10)	0x31	*Binary Offset*	*Description*	*Data Type*	*Units*
			0	1-Sigma Offset Correction Uncertainty (X)	float	meters
			4	1-Sigma Gyro Scale Factor Uncertainty	float	meters

				(Y)		
			8	1-Sigma Offset Correction Uncertainty (Z)	float	meters
			12	Valid Flags	U16	See Notes

MIP Packet Reference

Structure

Commands and Data are sent and received as fields in the MicroStrain "MIP" packet format. Below is the general definition of the structure:

Header				Payload			Checksum	
SYNC1 "u"	SYNC2 "e"	Descriptor Set byte	Payload Length byte	Fields			MSB	LSB
0x75	0x65	<desc set selector>	$k_1+k_2+\ldots k_n$	MIP Field 1 length = k_1	...	MIP Field *n* length = k_n	0x*MM*	0x*LL*

Field Header		Field Data
Field Length byte	Field Descriptor byte	Field Data
k_n	<descriptor>	<k_n-2 bytes of data>

The packet always begins with the start-of-packet sequence "ue" (0x75, 0x65). The "Descriptor Set" byte in the header specifies which command or data set is contained in fields of the packet. The payload length byte specifies the sum of all the field length bytes in the payload section.

Payload Length Range

Packet Header				Payload	Checksum	
SYN C1	SYN C2	Descript or Set	Payload Length	MIP Data Fields	MSB	LSB
				<--------------Payload Length Range --------------->		

The payload section can be empty or can contain one or more fields. Each field has a length byte and a descriptor byte. The field length byte specifies the length of the entire field including the field length byte and field descriptor byte. The descriptor byte specifies the command or data that is contained in the field data. The descriptor can only be from the set of descriptors specified by the descriptor set byte in the header. The field data can be anything but is always rigidly defined. The definition of a descriptor is fundamentally described in a ".h" file that corresponds to the definition of a descriptor set that the descriptor belongs to.

MicroStrain provides a “Packet Builder” functionality in the “MIP Monitor” software utility to simplify the construction of a MIP packet. Most commands will have a single field in the packet, but multiple field packets are possible. Extensive examples complete with checksums are given in the command reference section.

Checksum Range

The checksum is a 2 byte Fletcher checksum and encompasses all the bytes in the packet:

Packet Header				Payload	Checksum	
SYNC 1	SYNC 2	Descriptor Set	Payload Length	MIP Data Fields	MSB (byte1)	LSB (byte2)
<--- *Checksum Range* -->						

16-bit Fletcher Checksum Algorithm (C language)

```
for(i=0; i<checksum_range; i++)
 {
  checksum_byte1 += mip_packet[i];
  checksum_byte2 += checksum_byte1;
 }

checksum = ((u16) checksum_byte1 << 8) + (u16) checksum_byte2;
```

Advanced Programming

Multiple Commands in a Single Packet

MIP packets may contain one or more individual commands. In the case that multiple commands are transmitted in a single MIP packet, the 3DM-GQ4-45 will respond with a single packet containing multiple replies. As with any packet, all commands must be from the same descriptor set (you cannot mix Base commands with 3DM commands in the same packet).

Below is an example that shows how you can combine the commands from step 2 and 3 of the Example Setup Sequence into a single packet. The commands are from the 3DM set. The command packet has two fields as does the reply packet (the fields are put on separate rows for clarity):

Step 2 and 3	MIP Packet Header				Command/Reply Fields			Checksum	
	Sync1	*Sync2*	*Desc Set*	*Payload Length*	*Field Length*	*Cmd Desc.*	*Field Data*	*MSB*	*LSB*
Command field 1 Set IMU Message Format	**0x75**	**0x65**	**0x0C**	**0x14**	**0x0A**	**0x08**	Function: **0x00** Desc count: **0x02** 1st Descriptor: **0x03** Rate Dec: **0x000A** 2nd Descriptor:**0x04** Rate Dec: **0x000A**		
Command field 2 Set GNSS Message Format					**0x0A**	**0x09**	Function: **0x00** Desc Count: **0x02** ECEF posdesc: **0x04** Rate dec:**0x0004** ECEF veldesc: **0x06** Rate dec: **0x0004**	**0x50**	**0x98**
Replyfield 1 ACK/NACK	**0x75**	**0x65**	**0x0C**	**0x08**	**0x04**	**0xF1**	Cmd echo: **0x08** Error code: **0x00**		
Reply field 2 ACK/NACK					**0x04**	**0xF1**	Cmd echo: **0x09** Error code: **0x00**	**0xE9**	**0x6F**

Copy-Paste version of the command: "7565 0C14 0A08 0002 0300 0A04 000A 0A09 0002 0400 0406 0004 5098"

Note that the only difference in the packet headers of the single command packets compared to the multiple command packets is the payload length. Parsing multiple fields in a single packet involves subtracting the field length of the next field from the payload length until the payload length is less than or equal to zero.

Direct Modes

The 3DM-GQ4-45 has special "direct" modes that switch the device into an Sensor direct or GNSS direct device. The Device Communications Mode command is used to switch between modes. When in these modes, the 3DM-GQ4-45 acts like an "IMU only" or a u-blox GNSS sensor respectively. Any code or tools developed for these devices may be used in these modes. For example, when in the "u-blox" direct mode, the u-blox "u-center" application works perfectly with the GNSS chip embedded in the 3DM-GQ4-45.

These modes can be used to access advanced (native) data of the individual sensors, data that isn't represented in the 3DM command sets of the 3DM-GQ4-45. These modes are primarily advanced modes for programmers to allow the 3DM-GQ4-45 to be used in unusual situations where the normal functions of the GQ4-45 are bypassed.

IMPORTANT: When you switch modes, you are switching to a new device protocol EXCEPT for two commands: the Device Communications Mode and Device Status commands. Those commands are always available regardless of which mode you are in. For example, if you switch to GNSS direct mode, then the protocol recognized by the device is NMEA and UBX protocol, however the 3DM-GQ4-45 is still "listening" for mode switch or device status commands and will respond to them. It will not respond to any other 3DM-GQ4-45 Base or 3DM commands until switched back to the "Standard Mode".

IMPORTANT: The GNSS message settings required for Estimation Filter execution are automatically reloaded when switching from direct modes back in to standard mode.

Internal Diagnostic Functions

The 3DM-GQ4-45 supports two device specific internal functions used for diagnostics and system status. These are Device Built In Test and Device Status. These commands are defined generically but the implementation is very specific to the hardware implemented on this device. Other MicroStrain devices will have their own implementations of these functions depending on the internal hardware of the devices.

3DM-GQ4-45 Internal Diagnostic Commands

- Device Built In Test (0x01, 0x05)
- Device Status (0x0C, 0x64)

Handling High Rate Data

The size of the data fields from an inertial device is substantially greater than on most other types of sensors. On top of that, in many applications it is desirable to receive that data with the lowest latency possible and thus the highest BAUD rate is selected. The result is that the port servicing requirements in terms of both speed and buffer size can be surprisingly large for inertial data. This can lead to a couple of common problems: runaway latency and dropped packets.

Runaway latency

Most operating systems provide drivers that have ample buffers and take care of port servicing at the hardware level. Dropping packets or losing data is not usually an issue on these systems. What can be an issue is latency, that is, when the buffer is not emptied by the application in a timely manner. In the worst case, the buffer is being filled faster than it is emptied and the application operates with increasingly "old" data – which causes runaway latency. It is important to monitor the incoming data buffer to make sure you do not reach this condition.

Dropped packets

Many applications do not use an operating system but are written from scratch or on top of proprietary application frameworks. These are most often embedded MCUs or small single board microcontrollers. On these systems, port handling is usually done in code at the hardware level. Collecting data from a port requires the use one of three techniques: register polling, hardware interrupts, or direct memory access (DMA). Register polling is very easy to do and is adequate for simple communications where data comes in very small chunks and at reasonable data rates. The problem with register polling is that you either waste time looping while waiting for a byte to come in at the port or you get too busy doing other tasks so that by the time you poll the port, the byte is lost because the next one overwrites it. This causes dropped packets. On these systems, it is imperative to utilize either a hardware interrupt or hardware DMA on the UART receiving data from the 3DM-GQ4-45. The DMA or UART interrupt service routine only takes processor time when a byte is ready and as long as the interrupts are preemptive, the processor will fetch every byte received. Using the interrupt routine to fill a ring buffer makes the most efficient use of an MCU and makes it easier to write your application main line code. This is essentially what drivers in operating systems do.

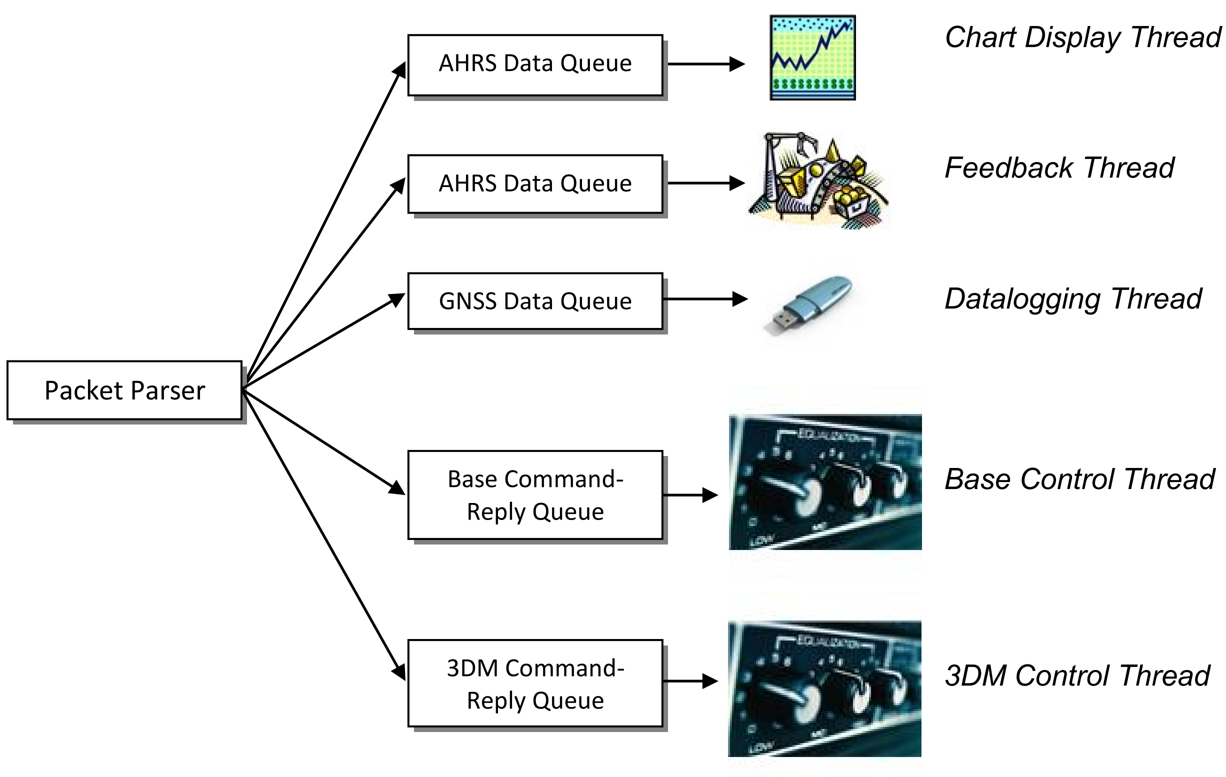
Creating Fixed Data Packet Format

The MIP packet structure and protocol provides a great deal of flexibility to the user for creating a custom data stream. It does this by allowing selectable data fields and individual data rates for each field. The side effect of this feature is that packets vary in size depending on what data is being delivered in any particular time frame. For example, if acceleration data is configured for 100Hz and magnetometer data is configured for 25Hz, every fourth packet is larger than the previous three because of the additional magnetometer data. In some applications, this is undesirable and there may be a requirement for a fixed packet structure so that each data packet is exactly the same. A fixed packet structure allows you to find data fields by fixed offsets rather than parsing the packet for each field.

A fixed packet structure is easily achieved with MIP packet protocol by simply making sure the data rate for each data quantity is the same. The order of the data fields in the packet reflect the order of the fields in the message format command and thus are completely under the control of the user. Once an acceptable data packet structure is determined, and all the rates are set to the same decimation, use the "Save current settings as startup settings" function selector in the message format command, and that format will be saved and used automatically on subsequent device startups. The message formats for each of the data classes (IMU, GNSS, EF, etc) work the same way, however the available data rates for each class is different, so you will need to create a fixed message format for each one.

Advanced Programming Models

Many applications will only require a single threaded programming model which is simple to implement using a single program loop that services incoming packets. In other applications, advanced techniques such as multithreading or event based processes are required. The MIP packet design simplifies implementation of these models. It does this by limiting the packet size to a maximum of 261 bytes and it provides the “descriptor set” byte in the header. The limited packet size makes scalable packet buffers possible even with limited memory space. The descriptor set byte aids in sorting an incoming packet stream into one or more command-reply packet queues and/or data packet queues. A typical multithreaded environment will have a command/control thread and one or more data processing threads. Each of these threads can be fed with individual incoming packet queues, each containing packets that only pertain to that thread – sorted by descriptor set. Packet queues can easily be created dynamically as threads are created and destroyed. All packet queues can be fed by a single incoming packet parser that runs continuously independent of the queues. The packet queues are individually scaled as appropriate to the process; smaller queues for lower latency and larger queues for more efficient batch processing of packets.



Multithreaded application with multiple incoming packet queues

End of Document

8500-0058 revision B

LORD Corporation
MicroStrain® Sensing Systems
459 Hurricane Lane,
Suite 102
Williston, VT 05495 USA
www.microstrain.com

ph: 800-449-3878
fax: 802-863-4093
sensing_support@LORD.com
sensing_sales@LORD.com

LORD MicroStrain® SENSING SYSTEMS