3-Space Sensor

3-Space Sensor Data-Logging

Miniature Attitude & Heading Reference System

User's Manual

Yost Labs
630 Second Street
Portsmouth, Ohio 45662

www.YostLabs.com

3-Space Sensor Data-Logging

Miniature Attitude & Heading Reference System

User's Manual

Yost Labs
630 Second Street
Portsmouth, Ohio 45662

www.YostLabs.com

Phone: 740-876-4936

Table of Contents

1. Usage/Safety Considerations

1.1 Usage Conditions

- Do not use the 3-Space Sensor in any system on which people's lives depend(life support, weapons, etc.)
- Because of its reliance on a compass, the 3-Space Sensor will not work properly near the earth's north or south pole.
- Because of its reliance on a compass and accelerometer, the 3-Space Sensor will not work properly in outer space or on planets with no magnetic field.
- Care should be taken when using the 3-Space Sensor in a car or other moving vehicle, as the disturbances caused by the vehicle's acceleration may cause the sensor to give inaccurate readings.
- Because of its reliance on a compass, care should be taken when using the 3-Space Sensor near ferrous metal structures, magnetic fields, current carrying conductors, and should be kept about 6 inches away from any computer screens or towers.
- Since the Data-Logging 3-Space Sensor uses removable MicroSD media, it is the end-user's responsibility to ensure that storage media is compatible.
- The Data-Logging 3-Space Sensor is powered by a rechargeable lithium-polymer battery. Lithium-polymer batteries have high energy densities and can be dangerous if not used properly. See section 1.4 Battery Considerations for further information pertaining to battery safety.

1.2 Technical Support and Repairs

Standard Limited Product Warranty: Yost Labs warrants the media and hardware on which products are furnished to be free from defects in materials and workmanship under normal use for sixty (60) days from the date of delivery except for OEM warranty items(see below). Yost Labs will repair or replace any defective product which is returned within this time period. Returned items will be tested in order to confirm a manufacturing defect is present. No warranties exist for any misuse.

OEM Limited Product Warranty: The following OEM products are subject to additional return limitations beyond the Standard Limited Product Warranty: surface-mount modules, integrated circuits, bare PCB modules, and other electronic components. Because of the risk of damage or malfunction due to user testing and handling problems, returns will be granted only upon evidence and/or inspection conclusively demonstrating manufacturing defect. All OEM products are individually tested prior to shipment for quality control.

Product Support: Yost Labs provides technical and user support via our toll-free number (740-876-4936) and via email (support@yostlabs.com). Support is provided for the lifetime of the equipment. Requests for repairs should be made through the Support department. For damage occurring outside of the warranty period or provisions, customers will be provided with cost estimates prior to repairs being performed.

1.3 Battery Safety Considerations

The Data-logging 3-Space Sensor contains a rechargeable lithium-polymer battery. Lithium-polymer batteries have high energy densities and can be dangerous if not used and cared for properly. The Data-Logging 3-space Sensor has been designed to include multiple levels of battery safety assurance. The Data-Logging 3-Space Sensor circuitry includes smart charging circuitry with thermal management to prevent over-charging the battery. The battery pack itself also includes protection circuitry to prevent over-charge, over-voltage, over-current, and over-discharge conditions.

Most battery issues arise from improper handling of batteries, and particularly from the continued use of damaged batteries.

As with any lithium-polymer battery-powered device, the following should be observed:

- Don’t disassemble, crush, puncture, shred, or otherwise attempt to change the form of your battery.
- Don't attempt to change or modify the battery yourself. Contact technical support for battery replacement or battery repair.
- Don’t let the mobile device or battery come in contact with water or other liquids.
- Don’t allow the battery to touch metal objects.
- Don’t place the sensor unit near a heat source. Excessive heat can damage the sensor unit or the battery. High temperatures can cause the battery to swell, leak, or malfunction.
- Don’t dry a wet or damp sensor unit with an appliance or heat source, such as a hair dryer or microwave oven.
- Don't drop the sensor unit. Dropping, especially on a hard surface, can potentially cause damage to the sensor unit or the battery.
- Discontinue use immediately and contact technical support if the battery or sensor unit produce odors, emit smoke, exhibit swelling, produce excess heat, exhibit leaking.
- Dispose of Lithium-polymer batteries properly in accordance with local, state , and federal guidelines.

2. Overview of the 3-Space Sensor

2.1 Introduction

The 3-Space Sensor™ Data-Logging integrates a miniature, high-precision, high-reliability, Attitude and Heading Reference System (AHRS) with a MicroSD card interface and a rechargeable lithium-polymer battery solution into a single low-cost end-use-ready unit. The Attitude and Heading Reference System (AHRS) uses triaxial gyroscope, accelerometer, and compass sensors in conjunction with advanced on-board filtering and processing algorithms to determine orientation relative to an absolute reference orientation in real-time.

Orientation can be returned in absolute terms or relative to a designated reference orientation. The proprietary multi-reference vector mode increases accuracy and greatly reduces and compensates for sensor error. The 3-Space Sensor Data-Logging system also utilizes a dynamic sensor confidence algorithm that ensures optimal accuracy and precision across a wide range of operating conditions.

The 3-Space Sensor™ Data-Logging unit features are accessible via a well-documented open communication protocol that allows access to all available sensor data and configuration parameters using a USB 2.0 interface, and configuration parameters are also accessible through configuration files on the SD card. Versatile commands allow access to raw sensor data, normalized sensor data, and filtered absolute and relative orientation outputs in multiple formats including: quaternion, Euler angles (pitch/roll/yaw), rotation matrix, axis angle, two vector(forward/up).

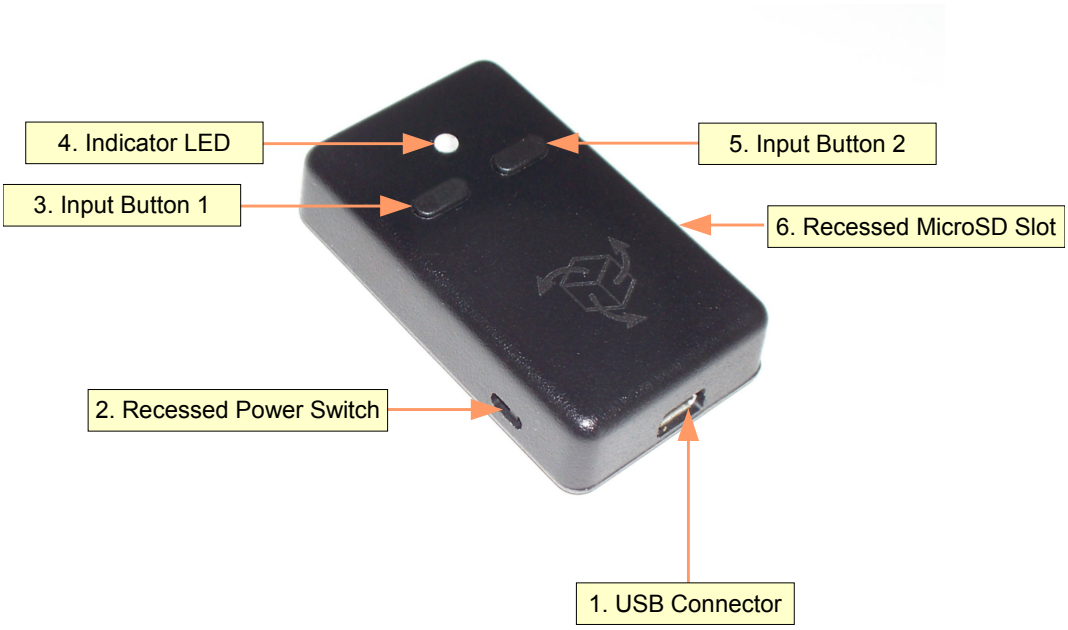
The 3-Space Sensor™ Data-Logging grants access to its SD card on a host PC through a USB Mass Storage interface. Settings for how to gather data can be accessed through configuration files, and gathered data can be stored in a variety of formats onto the SD card. The SD card uses the FAT32 filesystem.

When used as a USB device, the 3-Space Sensor™ provides mouse emulation and joystick emulation modes that ease integration with existing applications.

2.2 Applications

- Robotics performance analysis
- Motion capture
- Information gathering
- Personnel / pedestrian tracking
- Unmanned air/land/water vehicle tracking
- Education and performing arts
- Healthcare monitoring
- Asset tracking
- Vibration analysis and monitoring
- Event detection and monitoring

2.3 Hardware Overview



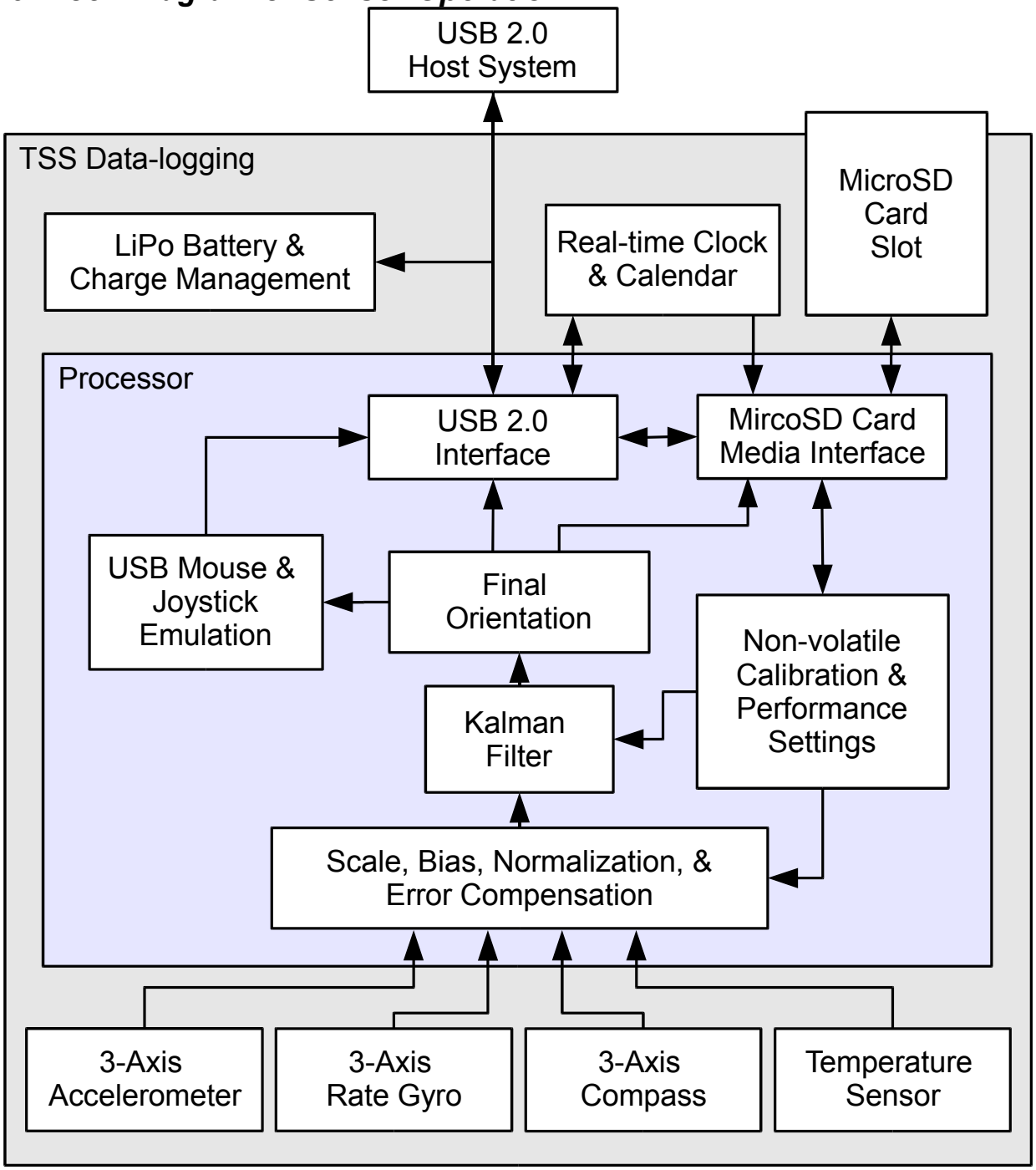
1. **USB Connector** – The 3-Space Sensor uses a 5-pin mini USB connector to connect to a computer via USB and to charge the internal battery. The USB connector provides for both power and communication signals.
2. **Recessed Power Switch** – The 3-Space Sensor can be switch on and off when powered from the internal battery by using the recessed power switch. When connected via USB, the unit is powered and the batteries will begin recharging regardless of the position of the recessed power switch
3. **Input Button 1** – The 3-Space Sensor includes two input buttons that can be used in conjunction with the orientation sensing capabilities of the device. The inputs are especially useful when using the 3-Space Sensor as an input device such as in joystick emulation mode or mouse emulation mode.
4. **Indicator LED** – The 3-Space Sensor includes an RGB LED that can be used for visual status feedback.
5. **Input Button 2** – The 3-Space Sensor includes two input buttons that can be used in conjunction with the orientation sensing capabilities of the device. The inputs are especially useful when using the 3-Space Sensor as an input device such as in joystick emulation mode or mouse emulation mode.
6. **Recessed MicroSD Card Slot** – The 3-Space Sensor MicroSD media can be inserted or removed through the recessed slot. This slot is recessed to help prevent accidental card removal.

2.4 Features

The 3-Space Sensor Data-Logging has many features that allow it to be a flexible all-in-one solution for your orientation sensing needs. Below are some of the key features:

- Small self-contained high-performance data-logging AHRS at 35mm x 60mm x 15mm and 28 grams
- Integrated Lithium-Polymer battery and charge control allows battery life of 5+ hours at full performance
- Fast sensor update and filter rate allow use in highly dynamic applications, including motion capture, performance & motion analysis, and navigation
- Highly customizable orientation sensing with options such as tunable filtering, oversampling, and orientation error correction
- Advanced integrated Kalman filtering allows sensor to automatically reduce the effects of sensor noise and sensor error
- Robust open protocol allows commands to be sent in human readable form, or more quickly in machine readable form
- Orientation output format available in absolute or relative terms in multiple formats (quaternion, rotation matrix, axis angle, two-vector)
- Absolute or custom reference axes
- Access to raw sensor data
- MicroSD card allows for data-logging applications, USB allows for real-time applications
- MicroSD card uses standard FAT32 file-system
- Flexible data logging configuration allows customization of logged data and allows event-based and time-based logging options
- Built-in clock/calendar provides for fully time-stamped event logging at high resolution
- USB communication through a virtual COM port
- Enumeration as USB mass-storage device makes access to logged data easy
- USB joystick/mouse emulation modes ease integration with existing applications
- Upgradeable firmware
- RGB status LED, two programmable input buttons
- Available in either hand-held or strap-down packaging
- RoHS compliant

2.5 Block Diagram of Sensor Operation

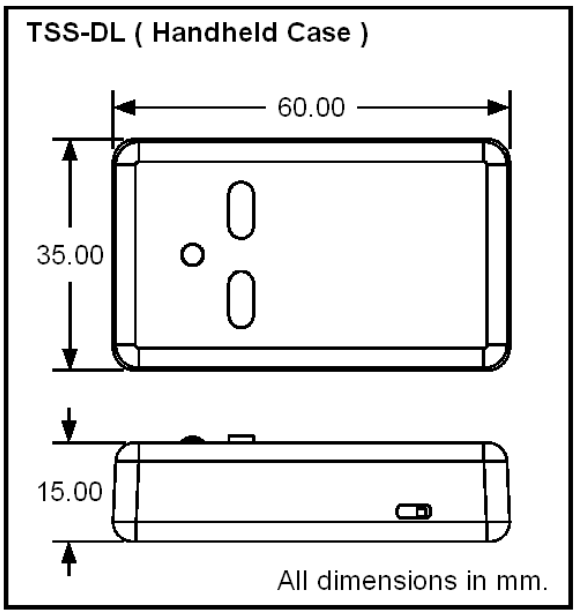


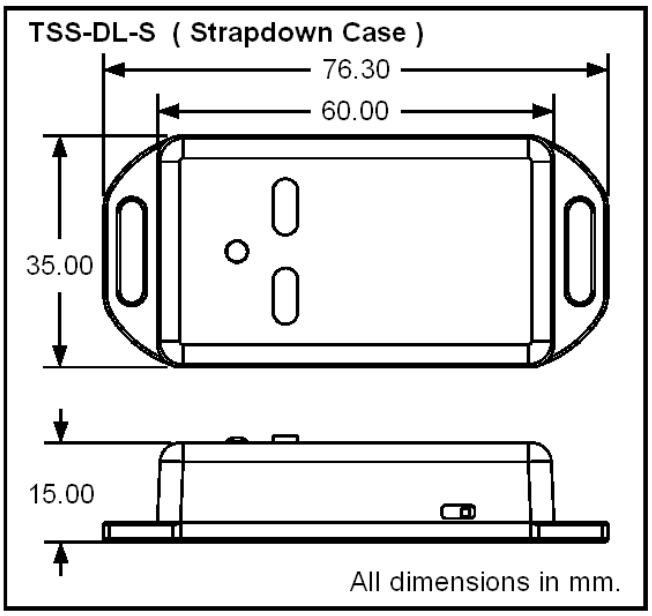
2.6 Specifications

General	
Part number	TSS-DL (Handheld Sensor Unit) TSS-DL-S (Strapdown Sensor Unit) TSS-DL-HH (High-range Handheld Sensor Unit) TSS-DL-HH-S (High-range Strapdown Sensor Unit)
Dimensions	35mm x 60mm x 15mm (1.38 x 2.36 x 0.59 in.)
Weight	28 grams (0.98 oz)
Supply voltage	+5v USB
Battery technology	rechargeable Lithium-Polymer
Battery lifetime	5+ hours continuous use at full performance
Communication interfaces	USB 2.0 (Mass Storage & CDC), MicroSD card configuration files.
Storage media	MicroSD card
Filter update rate	up to 200Hz with full functionality, up to 800Hz in IMU mode.
Orientation output	absolute & relative quaternion, Euler angles, axis angle, rotation matrix, two vector
Other output	raw sensor data, corrected sensor data, normalized sensor data, temperature, date/time.
Shock survivability	5000g
Temperature range	-40C ~ 85C (-40F ~ 185F)
Processor	32-bit RISC running @ 60MHz
Sensor	
Orientation range	360º about all axes
Orientation accuracy	±2º for dynamic conditions & all orientations
Orientation resolution	<0.08º
Orientation repeatability	0.085º for all orientations
Accelerometer scale	±2g / ±4g / ±8g selectable
Accelerometer resolution	14 bit
Accelerometer noise density	99µg/√Hz
Accelerometer sensitivity	0.00024g/digit for ±2g range (±6g range for HH) 0.00048g/digit for ±4g range (±12g range for HH) 0.00096g/digit for ±8g range (±24g range for HH)
Accelerometer temperature sensitivity	±0.008%/°C
Gyro scale	±250/±500/±2000 º/sec selectable
Gyro resolution	16 bit
Gyro noise density	0.03º/sec/√Hz
Gyro bias stability @ 25°C	11º/hr average for all axes
Gyro sensitivity	0.00875º/sec/digit for ±250º/sec 0.01750º/sec/digit for ±500º/sec 0.070º/sec/digit for ±2000º/sec
Gyro non-linearity	0.2% full-scale
Gyro temperature sensitivity	±0.016%/°C
Compass scale	±1.3 Ga default. Up to ±8.1 Ga available
Compass resolution	12 bit
Compass sensitivity	5 mGa/digit
Compass non-linearity	0.1% full-scale

*Specifications subject to change

2.7 Physical Dimensions





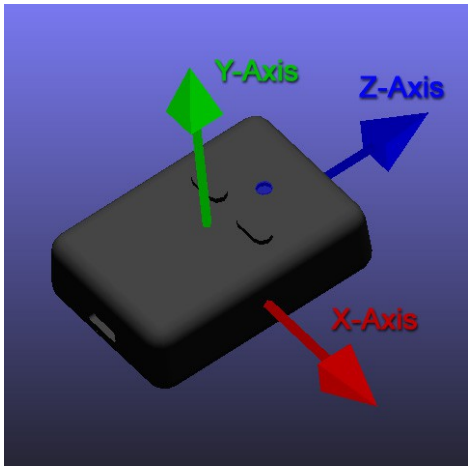
2.8 Axis Assignment

All 3-Space Sensor product family members have re-mappable axis assignments and axis directions. This flexibility allows axis assignment and axis direction to match the desired end-use requirements.

The natural axes of the 3-Space Sensor are as follows:

- The positive X-axis points out of the right hand side of the sensor, which is the side that is facing right when the buttons face upward and plug faces towards you.
- The positive Y-axis points out of the top of the sensor, the side with the buttons.
- The positive Z-axis points out of the front of the sensor, the side opposite the plug.

The natural axes are illustrated in the diagram below



Bear in mind the difference between natural axes and the axes that are used in protocol data. While they are by default the same, they can be remapped so that, for example, data axis Y could contain data from natural axis X. This allows users to work with data in a reference frame they are familiar with.

3. Description of the 3-Space Sensor

3.1 Orientation Estimation

The primary purpose of the 3-Space Sensor is to estimate orientation. In order to understand how to handle this estimation and use it in a meaningful way, there are a few concepts about the sensor that should be understood. The following sections describe these concepts.

3.1.1 Component Sensors

The 3-Space Sensor estimates orientation by combining the data it gets from three types of sensors: a gyroscope, an accelerometer, and a compass. A few things you should know about each of these sensors:

- **Accelerometer:** This sensor measures the acceleration due to gravity, as well as any other accelerations that occur. Because of this, this sensor is at its best when the 3-Space Sensor is sitting still. Most jitter seen as the orientation of the sensor changes is due to shaking causing perturbations in the accelerometer readings. To account for this, by default, when the 3-Space Sensor is being moved, the gyroscope becomes more trusted(becomes a greater part of the orientation estimate), and the accelerometer becomes less trusted.
- **Gyroscope:** This sensor measures angular motion. It has no ability to give any absolute orientation information like the accelerometer or compass, and so is most useful for correcting the orientation during sensor motion. Its role during these times becomes vital, though, as the accelerometer readings can become unreliable during motion.
- **Compass:** This sensor measures magnetic direction. The readings from the compass and accelerometer are used together to form the absolute component of orientation, which is used to correct any short term changes the gyroscope makes. Its readings are much more stable than those of the accelerometer, but it can be adversely affected by any ferrous metal or magnetic objects. When the accelerometer is less trusted, the compass is treated in the same way so as to avoid updates to orientation based on partial absolute information.

3.1.2 Scale, Bias, and Cross-Axis Effect

The readings taken from each component sensor are not in a readily usable form. The compass and accelerometer readings are not unit vectors, and the gyroscope readings aren't yet in radians per second. To convert them to these forms, scale and bias must be taken into account. Scale is how much larger the range of data read from the component sensor is than the range of data should be when it is converted. For example, if the compass were to give readings in the range of -500 to 500 on the x axis, but we would like it to be in the range of -1 to 1, the scale would be 500. Bias is how far the center of the data readings is from 0. If another compass read from -200 to 900 on the x axis, the bias would be 350, and the scale would be 550. The last parameter used in turning this component sensor data into usable data is cross-axis effect. This is the tendency for a little bit of data on one axis of a sensor to get mixed up with the other two. This is an effect experienced by the accelerometer and compass. There are 6 numbers for each of these, one to indicate how much each axis is affected by each other axis. Values for these are generally in the range of 1 to 10%. These parameters are applied in the following order:

1) Bias is added to each axis
2) The three axes are treated as a vector and multiplied by a matrix representing scale and cross-axis parameters

Factory calibration provides default values for these parameters for the accelerometer and compass, and users should probably never need to change these values. To determine these parameters for the gyroscope, you must calibrate it. Read the Quick Start guide or the 3-Space Suite manual for more information on how to do this.

3.1.3 Component Sensor Data Types

Component sensor data is presented by the 3-Space Sensor in three different stages and is readily accessible via certain protocol commands.

- **Raw Sensor Data:** This refers to data that is read directly from each of the component sensors before any additional processing has occurred. This kind of data is well-suited for users who wish to perform their own calibration routines as well as applications where precise analysis of motion is not extremely critical. Raw data commands are listed in Section 4.4.5, “Raw Data Commands” and span commands 0x40 through 0x43.

  Example: In the ±2G range, a raw accelerometer vector might look like (144, -25904, 744). This would indicate a force that is mostly in a downward direction.

- **Corrected Sensor Data:** This refers to 'raw' data that has been biased and scaled to represent real-world units, using the steps as described in Section 3.1.2, “Scale, Bias and Cross-Axis Effect”. There is an additional scaling that occurs, which further alters the data reading based on each component sensor's device-specific values. This scaling provides the real-world equivalents for read data. For the accelerometer, these values are in units of g-forces, for the magnetometer, these values are in units of gauss, and for the gyroscope, these values are in units of radians/sec. This kind of data is well-suited for users who wish to accurately track the motion of objects in 3D space or measure the strength and direction of magnetic fields. Corrected data commands are listed in Section 4.4.3, “Corrected Data Commands” and span commands 0x25 through 0x28.

  Example: In the ±2G range, the same raw accelerometer vector from before, when corrected, might look like (.004, -.791, .023). Note that these values are in units of g, and would indicate that at the moment of the sample, the sensor is accelerating mostly downwards at a rate of 7.75 meters per second squared.

- **Normalized Sensor Data:** This refers to 'corrected' data that has been geometrically normalized. For the accelerometer and magnetometer, all normalized sensor readings are unit-vectors and as such, have lengths of 1. For the gyroscope, these is no difference between 'corrected' and 'normalized' data. This kind of data is well-suited for users who are only interested in the direction of acceleration or magnetic fields. Normalized data commands are listed in Section 4.4.2, “Normalized Data Commands” and span commands 0x20 through 0x23.

  Example: The corrected accelerometer vector from before, when normalized, would look like (0.05, -0.998, 0.011). Note that the magnitude information is lost, and only the direction of the acceleration remains.

3.1.4 Additional Calibration

The 3-Space Sensor provides multiple calibration modes that can improve performance at the cost of additional setup and calibration routines. For more information on setting these additional modes, please refer to command 169.

- **Bias Mode**: Applies default range scaling to raw data readings. Also applies a bias offset to raw data, the values of which are taken from the provided calibration parameters command. (See section 4.3.7 for more information)
- **Bias / Scale Mode:** The default calibration mode. Applies default range scaling to raw data readings. Also applies a bias offset to the raw data as well as an additional scale matrix. Uses the matrix and vector portions from the provided calibration parameters command.
- **Ortho-Calibration Mode:** A more advanced calibration mode that requires initial setup steps (Please refer to the 3-Space Suite Quick Start Guide for information on how to supply ortho-calibration data) . Uses 24 orthogonal data points to provide accelerometer and compass correction factors for enhanced orientation accuracy.

3.1.5 Reference Vectors

In order to get an absolute estimation of orientation from the accelerometer and compass, the sensor needs a reference vector for each to compare to the data read from it. The most obvious choice for these are the standard direction of gravity(down) and the standard direction of magnetic force(north), respectively. However, the sensor does provide several different modes for determining which reference vector to use:

- **Single Manual:** Uses 2 reference vectors it is given as the reference vectors for the accelerometer and compass.
- **Single Auto:** When the sensor powers on or is put into this mode, it calculates gravity and north and uses those calculated vectors as the reference vectors.
- **Single Auto Continual:** The same as Single Auto, but the calculation happens constantly. This can account for some shifts in magnetic force due to nearby objects or change of location, and also can help to cope with the instability of the accelerometer.
- **Multiple:** Uses a set of reference vectors from which the best are picked each cycle to form a single, final reference vector. This mode has the ability to compensate for certain errors in the orientation. In this mode the sensor will have a slightly slower update rate, but will provide greater accuracy. For information on how to set up this mode, see the Quick Start guide or the 3-Space Suite manual.

3.1.6 Orientation Filtering

The 3-Space Sensor provides several different modes for providing orientation estimation. Note also that IMU data collection rate is bound to the update rate of the filter. For more information on setting these additional modes, please refer to command 123.

- **Kalman Filter:** The default filter mode. Normalized sensor data and reference vectors are fed into the Kalman filter, which uses statistical techniques to optimally combine the data into a final orientation reading. Provides the highest-accuracy orientation at the lowest performance.
- **Alternating Kalman Filter:** Uses the same Kalman filter as before, but skips every other update step. Slightly less accurate than the Kalman filter, but faster.
- **Complementary Filter**: Fuses low-pass filtered accelerometer/compass data with high-pass filtered gyroscope data to provide an orientation estimate. Less accurate than any Kalman filtering techniques, but provides significantly higher performance.
- **Quaternion Gradient Descent Filter**: Utilizes gradient descent techniques to avoid the high computational overhead of Kalman-based filters. Provides high performance and high accuracy.
- **IMU Mode:** Performs no orientation filtering, but allows IMU data to be read at the maximum update rate of 800 Hz.

3.1.7 Tare Orientation

Given the results of the Kalman filter, the sensor can make a good estimation of orientation, but it will likely be offset from the actual orientation of the device by a constant angle until it has been given a reference orientation. This reference orientation tells the sensor where you would like its zero orientation to be. The sensor will always consider the zero orientation to be the orientation in which the plug is facing towards you and top(the side with buttons on it) facing up. The sensor must be given a reference orientation that represents the orientation of the sensor when it is in the position in which you consider the plug to be towards you and the buttons up. The act of giving it this reference orientation to the sensor is called taring, just as some scales have a tare button which can be pressed to tell the scale that nothing is on it and it should read zero. For instructions on doing this, refer to the Quick Start guide or 3-Space Suite manual.

3.1.8 Offset Orientation

There are many applications for which it will be necessary or convenient to mount the sensor at odd angles, but it may also be desired in these situations that orientations can be treated as though the sensor were mounted normally. For example, if the sensor were mounted on a sloped surface of a vehicle like a car hood, it would be helpful if the orientations could read as though the sensor was mounted in a way that more closely matched the overall orientation of the vehicle, which does not include that slope.

The feature the sensor has to deal with mounting differences is the offset quaternion. This offset allows the sensor to pretend it is mounted in any given orientation while being actually mounted in any other actual orientation. To help understand the relationship between filtered orientation, tare orientation, and offset orientation, this is how the orientations are used by the sensor:

$$orientation_{final} = orientation_{tare} * orientation_{filtered} * orientation_{offset}$$

There are several ways to use this feature. The simplest way is if you happen to know the quaternion that represents the offset you want applied to the orientation, you can send this to the sensor by way of command 21(0x15). There are also commands to allow for more automated offset setting. To use these commands, do the following:

1) Place the sensor as close as possible to the mounting point, but in an orientation aligned with the overall vehicle or device the sensor is being mounted on, or in the orientation that you would like the sensor to act like it is in.

2) Call command 22, which sets a hidden variable called the “base offset” which affects the operation of the “Offset with current orientation” command. This will record your desired orientation later. If you ever want to reset this base offset, use command 20(0x14).

3) Mount the sensor onto the vehicle or device as you intend to for the end application.

4) Call command 19(0x13), which will set the offset based on the difference between the current orientation and the base offset. After this command is called, the sensor should now be acting as though it were in the desired orientation.

5) Make sure to commit the sensor settings to keep this change. Note that the base offset is not committable, but the offset itself is committable.

It should be noted that while it may seem like the set axis directions command could be used for the same purpose, this feature is the preferred way to deal with alternate mountings, as the axis directions mode has no way to account for a mounting that isn't a 90 degree based orientation away from the standard orientation. In addition, the axis direction mode does not handle switching the Euler angles to account for a different mounting, while this feature does.

3.1.9 Other Estimation Parameters

The 3-Space Sensor offers a few other parameters to filter the orientation estimate. Please note that these only affect the final orientation and not the readings of individual component sensors.

- **Oversampling:** Oversampling causes the sensor to take extra readings from each of the component sensors and average them before using them to estimate orientation. This can reduce noise, but also causes each cycle to take longer proportional to how many extra samples are being taken.
- **Running Average:** The final orientation estimate can be put through a running average, which will make the estimate smoother at the cost of introducing a small delay between physical motion and the sensor's estimation of that motion.
- **Trust Values:** As mentioned earlier, by default the accelerometer and compass are trusted less than the gyros when the sensor is in motion. These values involve parameters, one for the accelerometer and one for the compass, that indicate how much these component sensors are to be trusted relative to the gyroscope. These values range from 0 to 1, with 1 being fully trusted and 0 will be not trusted at all. There is a minimum and maximum truth value for each of the accelerometer and compass. The minimum will be used while the sensor is in motion, and the maximum will be used while it is still. To disable this sort of behavior, set both truth values to the same value. Note that the QGrad filter has its own set of trust values that can only be read or set while the sensor is in QGrad filter mode.

3.2 Communication

Obtaining data about orientation from the sensor or giving values for any of its settings is done through the sensor's communication protocol. The protocol can be used through the USB port. A complete description of how to use this protocol is given in section 4 of this document. Also, you may instead use the 3-Space Suite, which provides a graphical method to do the same. To learn how to use this, read the 3-Space Suite manual. In addition, data-logging options allow the sensor to be configured to log data in certain formats, with various criteria for beginning and ending a data-logging session. These options and working with the data-logging system is covered in section 3.4 of this document.

3.2.1 Wired Streaming Mode

The default mode of communication for the 3-Space Sensor is a call and response paradigm wherein you send a command and then receive a response. The sensor also features a streaming mode where it can be instructed to periodically send back the response from a command automatically, without any further communication from the host. To activate the streaming mode, use the following steps:

1) **Set up the streaming to call the commands you want data from. First, figure out which commands you want data from. The following commands are valid for streaming:**

0(0x00), Read tared orientation as quaternion
1(0x01), Read tared orientation as euler angles
2(0x02), Read tared orientation as rotation matrix
3(0x03), Read tared orientation as axis angle
4(0x04), Read tared orientation as two vector
5(0x05), Read difference quaternion
6(0x06), Read untared orientation as quaternion
7(0x07), Read untared orientation as euler angles
8(0x08), Read untared orientation as rotation matrix
9(0x09), Read untared orientation as axis angle
10(0x0a), Read untared orientation as two vector
11(0x0b), Read tared two vector in sensor frame
12(0x0c), Read untared two vector in sensor frame
32(0x20), Read all normalized component sensor data
33(0x21), Read normalized gyroscope vector
34(0x22), Read normalized accelerometer vector
35(0x23), Read normalized compass vector
37(0x25), Read all corrected component sensor data
38(0x26), Read corrected gyroscope vector
39(0x27), Read corrected accelerometer vector
40(0x28), Read corrected compass vector
41(0x29), Read corrected linear acceleration
43(0x2B) Read temperature C
44(0x2C), Read temperature F
45(0x2D), Read confidence factor
64(0x40), Read all raw component sensor data
65(0x41), Read raw gyroscope vector
66(0x42), Read raw accelerometer vector
67(0x43), Read raw compass vector
201(0xc9), Read battery voltage
202(0xca), Read battery percentage
203(0xcb), Read battery status
250(0xfa), Read button state
255(0xff), No command

There are 8 streaming slots available for use, and each one can hold one of these commands. These slots can be set using command 80(0x50), with the parameters being the 8 command bytes corresponding to each slot. Unused slots should be filled with 0xff so that they will output nothing.

Please note: The total amount of data the 8 slots can return at once is 256 bytes. If the resulting data exceeds this, the set streaming slots command will fail.

2) **Set up the streaming interval, duration, and start delay.** These parameters control the timing of the streaming session. They can be set using command 82(0x52). All times are to be given in microseconds. They control the streaming as follows:

   Interval determines how often the streaming session will output data from the requested commands. For example, an interval of 1000000 will output data once a second. An interval of 0 will output data as quickly as possible. The interval will be clamped to 1000 if the user attempts to set it in the range 1 – 1000.

   Duration determines how long the streaming session will run for. For example, a duration of 5000000 indicates the session should stop after 5 seconds. A duration of 4294967295 (0xFFFFFFFF) means that the session will run indefinitely until a stop streaming command is explicitly issued.

   Start Delay determines how long the sensor should wait after a start command is issued to actually begin streaming. For example, a start delay 200000 means the session will start after 200 milliseconds.

3) **Begin the streaming session.** This can be done using command 85(0x55). Once started, the session will run until the duration has elapsed, or until the stop command, 86(0x56) has been called. Please note that only binary data is supported. While streaming sessions can be started with ascii commands, only binary data will be returned. Also note that if the sensor is sending large amounts of data the host doesn't have time to handle, this can cause buffer overflows in some communication drivers, leading to slowdowns and loss of data integrity. If the firmware detects that the buffer has overflowed, the asynchronous session will be stopped. If this occurs, this is a sure sign that either the streaming interval is set too low, the program is not working fast enough to handle the amount of data or both.

For more information on all these commands, see the Streaming Commands section in the command chart near the end of this document.

3.3 Input Device Emulation

3.3.1 Axes and Buttons

The 3-Space Sensor has the ability to act as a joystick and/or mouse. Both of these are defined in the same way, as a collection of axes and buttons. Axes are input elements that can take on a range of values, whereas buttons can only either be on or off. On a joystick, the stick part would be represented as 2 axes, and all the physical buttons on it as buttons. The 3-Space Sensor has no physical joystick and only 2 physical buttons, so there are a number of options to use properties of the orientation data as axes and buttons. Each input device on the 3-Space Sensor has 2 axes and 8 buttons. For more information on setting these up, see the 3-Space Suite manual. All communication for these input devices is done through the standard USB HID(Human Interface Device) protocol.

3.3.2 Joystick

As far as a modern operating system is concerned, a joystick is any random collection of axes and buttons that isn't a mouse or keyboard. Joysticks are mostly used for games, but can also be used for simulation, robot controls, or other applications. The 3-Space Sensor, as a joystick, should appear just like any other joystick to an operating system that supports USB HID(which most do).

3.3.3 Mouse

When acting as a mouse, the 3-Space Sensor will take control of the system's mouse cursor, meaning if the mouse portion is not properly calibrated, using it could easily leave you in a situation in which you are unable to control the mouse cursor at all. In cases like this, unplugging the 3-Space Sensor will restore the mouse to normal operation, and unless the mouse enabled setting was saved to the sensor's memory, plugging it back in should restore normal operation. Using the default mouse settings, caution should be exercised in making sure the orientation estimate is properly calibrated before turning on the mouse. For help with this, see the Quick Start guide.

The mouse defaults to being in Absolute mode, which means that the data it gives is meant to represent a specific position on screen, rather than an offset from the last position. This can be changed to Relative mode, where the data represents an offset. In this mode, the data which would have indicated the edges of the screen in Absolute mode will now represent the mouse moving as quickly as it can in the direction of that edge of the screen. For more information, see command 251 in section 4.3.7, or the 3-Space Suite manual.

3.4 Data-Logging

3.4.1 Mass Storage Device

The Data-Logging 3-Space Sensor exposes the contents of its SD card to a computer by enumerating as a Mass Storage device in addition to a virtual COM port. Upon being connected to a computer through USB, the sensor will cease any current data-logging session and will cede control of the SD card to the computer, as both the computer and the sensor cannot write to the SD card without coming into conflict. No further data-logging can be done at this point until the computer no longer controls the SD card. Unplugging the sensor will return control of the SD card to it. Also, the sensor has a Mass Storage Off mode which will return control of the SD card to it even while attached to a computer. For more information on this, see commands 57 and 58 in section 4.

3.4.2 SD Card Format and Directory Structure

The Data-Logging 3-Space Sensor will attempt, upon power on or upon SD card insertion, to place the directory structure it requires on to the card. If this is unsuccessful for any reason, such as the card being in read only mode or the card having not been formatted yet, the sensor's LED will pulse red twice quickly once a second. It will also do this if no SD card is inserted at all. This indicates that it is not ready to do data-logging. The SD card may be formatted in one of two ways: either let any SD card formatting utility(such as the one built in to Windows) format the card as a single FAT32 partition, or insert the unformatted card into the Data-Logging 3-Space Sensor and call command 59(look in section 4 for details on this command). Also, the 3-Space Suite has an easy way to call this command in the Sensor Info window when connected to a Data-Logging sensor. Refer to the Data-Logging Quick Start guide for more information. When the card has been properly initialized by the sensor, the LED will turn solid blue. The directory structure the sensor sets up is as follows:

/ (Root Directory)

/Data/

/Config/

The **/Data/** directory is for holding the results of any data capture sessions. A new directory inside this will be created for each session, named according to when the capture started (For information on setting up the current time, see section 3.4.4).

This **/Config/** directory holds configuration files which can be modified to change the current settings of the sensor. It contains two files: **sensor.cfg** and **capture.cfg**. **sensor.cfg** allows access to many sensor settings that can also be modified through protocol commands, such as orientation averaging modes. **capture.cfg** allows access to settings having to do with how and how often data is gathered. See section 3.4.3 for information on the contents of capture.cfg and data-logging options in general. In order to change a setting in one of these files, simply find the name of the property you want to modify on the left of an equals sign, and then change the value for it on the right of the equals sign. If the value is textual (properties where this is the case will have a list of the possible values listed to the right of the assignment), be sure to enclose the text in quotes. Quotes are not necessary for numeric values.

3.4.3 Data-Logging

As described in section 3.4.2, upon the start of a data-logging session, a new directory will be created to hold the data, named after the time the session was started(given by the real time clock, see section 3.4.4).

The following sections describe the configuration properties that determine data-logging capture behavior.

CaptureStartEvent

The CaptureStartEvent property in the **capture.cfg** file selects options for starting a capture session. Possible values for the CaptureStartEvent property are:

- “on command”: This setting has no way of starting a capture session except through the calling of the begin data-logging session command, command 60. See section 4 for more information on this command. Because calling a command requires a USB connection which can communicate with the sensor, the sensor will have to

be taken out of Mass Storage mode before this command is called. The command to turn off Mass Storage mode is command 58. Also note that regardless of start event, this command can be used to start a data-logging session.

- "on startup": Whenever the sensor starts up, it will attempt to start logging data as soon as it can. After this one session, it will not attempt to start another session.
- "left button", "right button", and "both buttons": A session will begin when the appropriate button or buttons are pressed. Note that if the stop condition is set to the same set or part of the same set of buttons, the buttons will need to be released before they will register as a stop condition.
- "motion": A session will begin when the accelerometer detects that motion has risen to a certain level. This level is given in *g*s(gravity units) and can be set through the property CaptureStartEventMotionThreshold.

CaptureStopEvent

The CaptureStopEvent property in the **capture.cfg** file selects options for stopping a capture session. Possible values for the CaptureStopEvent property are:

- "on command": Like the same property for the start events, this means a session can only be stopped through a command. Use command 61 for ending a data-logging session. Also note that regardless of stop event, this command can be used to stop a data-logging session.
- "always": This will always stop a data-logging session as soon as it starts. This is most useful in concert with the CapturePostStopGatherTime property, which gives a length of time after the stop of a session data should continue to be logged.
- "left button", "right button", and "both buttons": Just as for start events, these will stop a session when the buttons are pressed.
- "motion stop": This will stop a session when the motion falls below a certain threshold. This threshold, given in *g*s(gravity units), is indicated by the property CaptureStopEventMotionThreshold.
- "capture count": This will stop a session after a certain number of samples have been taken. This number is given by the property CaptureStopEventCaptureCount.
- "capture duration": This will stop a session after it has lasted for a certain amount of time, given in milliseconds by the property CaptureStopEventCaptureDuration.
- "period count": This only has any effect when the CaptureStyle property is set to "periodic". It will stop a capture after a certain number of capture periods, given by the property CaptureStopEventPeriodCount. Using this when in continuous mode will cause the session to never end.

CaptureFormat

Th CaptureFormat property specifies a format string similar to that required by the C function printf. It consists of a string of whatever characters are desired to show up in the data-logging file, in addition to a number of % delimited tokens which indicate data of a certain type. Options for characters which may follow the % are as follows:

d: The current date.
t: The current time.
s: Timestamp in microseconds.
q: The tared orientation, in quaternion form.
x: The tared orientation, in axis angle form.
e: The tared orientation, in Euler angle form(given in pitch, yaw, roll order).
m: The tared orientation, in rotation matrix form.
g: The current calculated angular difference between readings, in quaternion form.
uq: The untared orientation, in quaternion form.
ua: The untared orientation, in axis angle form.
ue: The untared orientation, in Euler angle form(given in pitch, yaw, roll order).
um: The untared orientation, in rotation matrix form.
ng: The latest normalized(scaled and biased) gyroscope reading.
na: The latest normalized(scaled and biased) accelerometer reading.
nc: The latest normalized(scaled and biased) compass reading.

nt: The latest temperature reading, in degrees C.
nb: The battery level percentage.
cg: The corrected (in units of rad/sec) gyroscope reading.
ca: The corrected (in units of g) accelerometer reading.
cc: The corrected (in units of gauss) compass reading
rg: The raw(as it comes from the sensor) gyroscope reading.
ra: The raw(as it comes from the sensor) accelerometer reading.
rc: The raw(as it comes from the sensor) compass reading.
L: The corrected linear acceleration.
%: An actual %.

Note that any non-data characters will only be included in the file in ASCII data-logging mode. For more information, see the property CaptureDataMode. In ASCII mode, data that is comprised of multiple values will be separated by commas.

CaptureInterval

The CaptureInterval property in the **capture.cfg** file determines the desired sampling period while logging is active. This property is specified in milliseconds. Do note that it is possible to set this value to an interval that is faster than the SD card interface can easily handle, which will cause the sensor to drop some samples in order to keep up. If the desired effect is to sample as fast as possible, this value should be set to"auto", which indicates to the sensor that it should set the interval automatically based on the amount of data being captured, to reduce the chance of dropped samples.

CaptureStyle

The CaptureStyle property in the **capture.cfg** file determines if "continuous" or "periodic" data capture style is used. In continuous mode, data-logging will be enabled at all times during a data-logging session. In periodic mode, data-logging will start and stop during the course of a session. The properties that control this behavior are CaptureStylePeriodicCaptureTime, which determines how long it captures for before it stops, and CaptureStylePeriodicRestTime, which determines how long it is stopped before it starts capturing again.

CapturePostStopGatherTime

The CapturePostStopGatherTime property in the **capture.cfg** file specifies an amount of time, after a session has stopped, to continue gathering data. This value is specified in milliseconds.

CaptureFileStub

The CaptureFileStub property in the **capture.cfg** file specifies the base name of the data-logging file which resides in each session's directory, with a ".txt" appended to it if data is captured in ASCII mode, and ".dat" if in binary mode. With a CaptureFileMode of "new", this stub will also have a number added on to it before the ".txt" or ".dat".

CaptureFileMode

The CaptureFileMode property in the **capture.cfg** file specifies how files are used to capture data. This property is one of the following:

- "append": New samples are added to the end of the single data log in the session directory.
- "replace": In continuous mode, the same as append. In periodic mode, only data from the most recent period will appear.
- "new": A new file will be made for each data capture period, and each period's data will be placed in this new file. Only one file will be made in continuous mode, named "<CaptureFileStub>1.txt" in ASCII mode or "<CaptureFileStub>1.dat" in binary.

CaptureDataMode

The CaptureDataMode property in the **capture.cfg** file specifies whether captured data is stored in the file as human readable ASCII or compact binary. This property is one of the following:

- “ascii”: Data is logged in a human readable form. Superfluous characters in the formatting string are placed in the data log.
- “binary”: Data is logged in a compact, non-human readable form. Superfluous characters are ignored.

CaptureFileInfoHeader

The CaptureFileInfoHeader property in the **capture.cfg** file determines whether a line should be written at the top of each capture file indicating the format of the data contained within. This property can be set to either 0 for off or 1 for on.

3.4.4 LED Capture Behavior

The RGB LED can be used as an indicator of the current state of the sensor. Below is a summary of the LED meanings:

Solid Blue (or solid currently set custom color setting): Power on and no data-logging session in progress.

Yellow: A data-logging session is in progress, but a sample is not currently being taken.

Green: The LED will emit a green flash and return to yellow when a data-logging sample is taken.

Red (double pulse): MicroSD media not present or file system error.

Red (Single Pulse when not plugged into USB): Battery low - battery life remaining is at or below 5%.

Yellow (Single Pulse when plugged into USB): Battery is actively charging.

Green (Single Pulse when plugged into USB): Battery is fully-charged.

Red (Rapid Constant Pulse): Panic mode. Indicates corrupted sensor settings.

3.4.5 Real Time Clock

The Data-Logging 3-Space Sensor contains a real time clock chip which allows it to keep track of time. The clock chip uses a separate clock battery which maintains the time and clock settings. This internal clock battery is calculated to have a life of about 5 years. The clock chip must be given an initial time for it to report time properly in a desired time zone. This time can be given to the chip using command 62, and read back using command 63. See the entries for these commands in Section 4 for more details. In addition, the 3-Space Suite has an option for automatically setting the clock of the sensor to the clock of the host computer. Go to the Sensor Info window when connected to the Data-Logging sensor and there will be an option to do this. For more information, refer to the Data-Logging Quick Start Guide.

3.5 Sensor Settings

3.5.1 Committing Settings

Changes made to the 3-Space Sensor will not be saved unless they are committed. This allows you to make changes to the sensor and easily revert it to its previous state by resetting the chip. For instructions on how to commit your changes, see the Quick Start guide or 3-Space Suite manual. Any changes relating to the multiple reference vector mode are an exception to this rule, as all these changes are saved immediately.

3.5.2 Natural Axes

The natural axes of the 3-Space Sensor are as follows:

- The positive X-axis points out of the right hand side of the sensor, which is the side that is facing right when the buttons face upward and plug faces towards you.
- The positive Y-axis points out of the top of the sensor, the side with the buttons.
- The positive Z-axis points out of the front of the sensor, the side opposite the plug.

Bear in mind the difference between natural axes and the axes that are used in protocol data. While they are by default the same, they can be remapped so that, for example, data axis Y could contain data from natural axis X. This allows users to work with data in a reference frame they are familiar with.

See section 2.8 for a diagram illustrating the natural axes.

3.5.3 Sensor Settings and Defaults

Setting Name	Purpose	Default Value
Accelerometer Trust Values	Determine how trusted the accelerometer is	Minimum of 1/101, maximum of 1/6
Compass Trust Values	Determine how trusted the compass is	Minimum of 1/101, maximum of 1/6
Accelerometer Coefficients	Determines the scale, bias, and cross-axis parameters for the accelerometer	Factory calibrated
Compass Coefficients	Determines the scale, bias, and cross-axis parameters for the compass	Factory calibrated
Gyroscope Coefficients	Determines the scale, bias and cross-axis parameters for the gyroscope	Factory calibrated
Accelerometer Enabled	Determines whether the compass is enabled or not	TRUE
Compass Enabled	Determines whether the accelerometer is enabled or not	TRUE
Gyroscope Enabled	Determines whether the gyroscope is enabled or not	TRUE
Filter Mode	Determines how orientation is filtered.	1 (Kalman)
Accelerometer Reference Vector	Determines which vector the accelerometer should read in order for the sensor's untared orientation to be the identity orientation.	0, 1, 0
Compass Reference Vector	Dertemines which vector the compass should read in order for the sensor's untared orientation to be the identity orientation.	0, 0, 1 (Default mode is to re-calculate this vector on startup)
Reference Vector Mode	Determines how reference vectors are calculated for orientation estimation.	1 (Single automatic)
Euler Order	Determines the default composition order of euler angles returned by the sensor.	YXZ
Calibration Mode	Determines how raw sensor data is transformed into normalized data	1 (Scale-Bias)
Axis Directions	Determines what natural axis direction each data axis faces	+X, +Y, +Z
Sample Rate	Determines how many samples the sensor takes per cycle	1 from each component sensor
Running Average Percentage	Determines how heavy of a running average to run on the final orientation	0(no running average)
Desired Update Rate	Determines how long each cycle should take(ideally)	0 microseconds
RS232 Baud Rate	Determines the speed of RS232 communication	115200
CPU Speed	Determines how fast the CPU will run	60 MHz

LED Color	Determines the RGB color of the LED	0,0,1(Blue)
Joystick Enabled	Determines whether the joystick is enabled or not	TRUE
Mouse Enabled	Determines whether the mouse is enabled or not	FALSE
Button Gyro Disable Length	Determines how many cycles the gyro is ignored after a button is pressed	5
Multi Reference Weight Power	Determines what power each multi reference vector weight is raised to	10
Multi Reference Cell Divisions	Determines how many cells the multi reference lookup table is divided into per axis	4
Multi Reference Nearby Vectors	Determines how many nearby vectors each multi reference lookup table cell stores	8
Wired Response Header Bitfield	Determines what kind of data is prepended to response data.	0
Streaming Slots	Determines which commands are executed during a streaming session.	255, 255, 255, 255, 255, 255, 255, 255
Streaming Timing	Dertemines the streaming interval, duration and delay.	10000, 4294967295, 0

3.5.4 Capture Settings and Defaults

Setting Name	Purpose	Default Value
Capture Rate	Determine how often to capture data(in ms)	“auto”
Capture Format	Determine what data to capture	“%d %t %q”(Date, time, and orientation as quaternion)
Start Event	Determine how a session is started	“left button”
Start Event Motion Threshold	Determine how much motion starts a session(in *g*)	0.5
Capture Style	Determine when to capture data	“continuous”
Periodic Capture Time	Determine how long to capture data in a period(in ms)	5000
Periodic Rest Time	Determine how long to rest in a period(in ms)	5000
Stop Event	Determine how a session is stopped	“right button”
Stop Event Motion Threshold	Determine how much motion stops a session(in *g*)	0.3
Stop Event Capture Count	Determine how many captures to perform before stop	100
Stop Event Capture Duration	Determine how long to capture before stop(in ms)	5000
Stop Event Period Count	Determine how many periods before stopping	1
Post Stop Gather Time	Determine how long to capture after stop	0
File Stub	Determine name of data log	“data”
File Mode	Determines how to put data in files	“append”
Data Mode	Determines how data is written	“ascii”
File Info Header Enabled	Determines whether or not to have a file header	1

4. 3-Space Sensor Usage/Protocol

4.1. Usage Overview

4.1.1 Protocol Overview

The 3-Space Sensor receives messages from the controlling system in the form of sequences of serial communication bytes called packets. For ease of use and flexibility of operation, two methods of encoding commands are provided: binary and text. Binary encoding is more compact, more efficient, and easier to access programmatically. ASCII text encoding is more verbose and less efficient yet is easier to read and easier to access via a traditional terminal interface. Both binary and ASCII text encoding methods share an identical command structure and support the entire 3-Space command set.

The 3-Space Sensor buffers the incoming command stream and will only take an action once the entire packet has been received and the checksum has been verified as correct(ASCII mode commands do not use checksums for convenience). Incomplete packets and packets with incorrect checksums will be ignored. This allows the controlling system to send command data at leisure without loss of functionality. The command buffer will, however, be cleared whenever the 3-Space Sensor is either reset or powered off/on.

Specific details of the 3-Space Sensor protocol and its control commands are discussed in the following pages.

4.1.2 Computer Interfacing Overview

The Data-logging 3-Space Sensor enumerates as a composite USB device consisting of a USB mass storage device, a CDC serial communication device, and an HID mouse/keyboard device.

When interfacing with a computer, the 3-Space Sensor presents itself as a COM port, which provides an interface by which the serial communication the protocol requires may happen. The name of this COM port is specific to the operating system being used. It is possible to use multiple 3-Space Sensors on a single computer. Each will be assigned its own COM port.

The USB mass storage device allows access the configuration files that are used to determine sensor configuration and data-logging options. For more detail on these files refer to section 3.4 Data-Logging.

For more information on how to install the sensor software on a computer and begin using it, see the Quick Start guide.

4.2. Protocol Packet Format

4.2.1 Binary Packet Format

The binary packet size can be three or more bytes long, depending upon the nature of the command being sent to the controller. Each packet consists of an initial **"start of packet"** byte, followed by a **"command value"** specifier byte, followed by zero or more "**command data"** bytes, and terminated by a packet "**checksum value**" byte.

Each binary packet is at least 3 bytes in length and is formatted as shown in figure 1

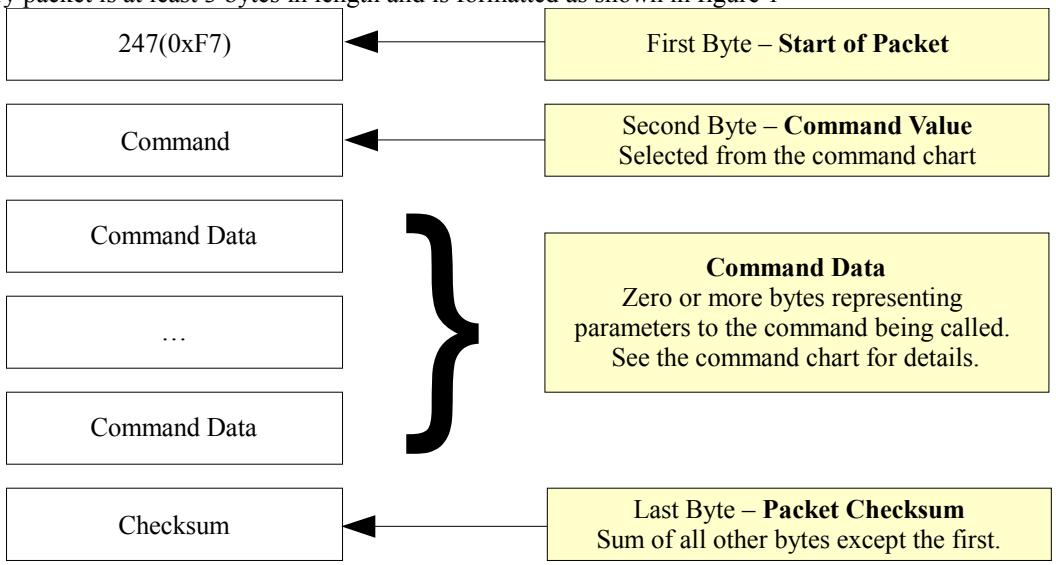


Figure 1 - Typical Binary Command Packet Format

Binary Return Values:

When a 3 Space Sensor command is called in binary mode, any data it returns will also be in binary format. For example, if a floating point number is returned, it will be returned as its 4 byte binary representation.

For information on the floating point format, go here: http://en.wikipedia.org/wiki/Single_precision_floating-point_format

Also keep in mind that integer and floating point values coming from the sensor are stored in big-endian format.

The Checksum Value:

The checksum is computed as an arithmetic summation of all of the characters in the packet (except the checksum value itself) modulus 256. This gives a resulting checksum in the range 0 to 255. The checksum for binary packets is transmitted as a single 8-bit byte value.

4.2.2 ASCII Text Packet Format

ASCII text command packets are similar to binary command packets, but are received as a single formatted line of text. Each text line consists of the following: an ASCII colon character followed by an integral command id in decimal, followed by a list of ASCII encoded floating-point command values, followed by a terminating newline character. The command id and command values are given in decimal. The ASCII encoded command values must be separated by an ASCII comma character or an ASCII space character. Thus, legal command characters are: the colon, the comma, the period, the digits 0 through 9, the minus sign, the new-line, the space, and the backspace. When a command calls for an integer or byte sized parameter, the floating point number given for that parameter will be interpreted as being the appropriate data type. For simplicity, the ASCII encoded commands follow the same format as the binary encoded commands, but ASCII text encodings of values are used rather than raw binary encodings.

Each ASCII packet is formatted as shown in figure 2.

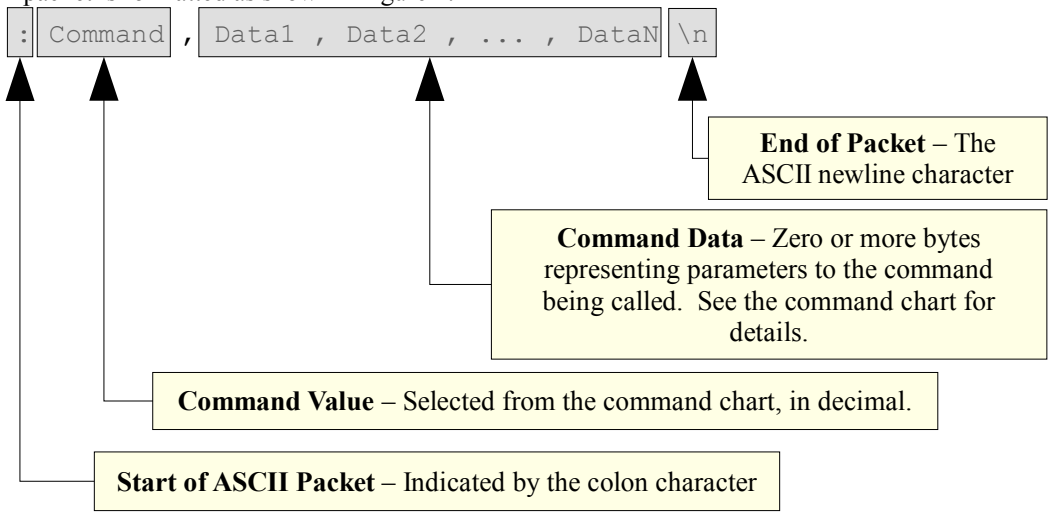


Figure 2 - Typical ASCII Command Packet Format

Thus the ASCII packet consists of the the following characters:

- **:** – the ASCII colon character signifies the start of an ASCII text packet.
- **,** – the ASCII comma character acts as a value delimiter when multiple values are specified.
- **.** – the ASCII period character is used in floating point numbers.
- **0~9** – the ASCII digits are used to in integer and floating point values.
- **-** the ASCII minus sign is used to indicate a negative number
- **\n** – the ASCII newline character is used to signify the end of an ASCII command packet.
- **\b** – the ASCII backspace character can be used to backup through the partially completed line to correct errors.

If a command is given in ASCII mode but does not have the right number of parameters, the entire command will be ignored.

Sample ASCII commands:

`:0\n`	Read orientation as a quaternion
`:106,2\n`	Set oversample rate to 2

ASCII Return Values:

All values are returned in ASCII text format when an ASCII-format command is issued. To read the return data, simply read data from the sensor until a Windows newline(a carriage return and a line feed) is encountered..

4.3 Response Header Format

4.3.1 Wired Response Header

The 3-Space Sensor is capable of returning additional data that can be prepended to all command responses. This capability is managed via the Response Header Bitfield, which can be configured using command 221 (0xDD). Each bit in the field, if enabled, corresponds to a different piece of information that will be output prior to the expected response data. To use the Response Header Bitfield, use the following steps:

1.) **Determine which additional data you would like to have output as the response header. The list of options are:**

0x1 (Bit 0) – Success/Failure; comprised of one byte with non-zero values indicating failure.

0x2 (Bit 1) – Timestamp; comprised of four bytes representing the most recent sample time in microseconds. Note that this is not a difference, but a total accumulated time.

0x4 (Bit 2) – Command echo; comprised of one byte. Echoes back the previous command.

0x8 (Bit 3) – Additive checksum; comprised of one byte summed over the response data modulus 256. Note that this does not include the Response Header itself.

0x10 (Bit 4) – Logical ID; comprised of one byte indicating the logical ID of the received packet. For wired communication, this always returns 0xFE.

0x20 (Bit 5) – Serial number; comprised of four bytes.

0x40 (Bit 6) – Data length; comprised of one byte. Represents the amount of response data. Note that this does not include the Response Header itself.

For example, if you wanted all future data to be preceded with a timestamp and a data length, you would want to use bits 1 and 6, which corresponds to the value 66 (0x00000042). This is the value that would be passed into the Set Wired Response Header Bitfield command (Command 221).

2.) **Call command 221 passing in the specified value. Keep in mind that this is a 4-byte value.**

3.) **Ask for data using the Response Header Start Byte.**

Typical wired binary commands use 0xF7 to indicate the start of a command packet. If 0xF7 is used, response data will never contain a Response Header. Instead, the user should use 0xF9 instead of 0xF7. This will cause the resulting command to prepend the requested Response Header to the response data. Typical wired ascii commands use ':' to indicate the start of a typical command packet and the ';' character to indicate to the sensor that the data should have the Response Header prepended. Also note that all Response Header will be output in ascending order, starting with the lowest enabled bit and continuing on to the highest enabled bit.

4.) **Parse the Response Header data.**

Assume we wanted to ask for the raw accelerometer data along with the timestamp and data length and that we have already called command 221 with a parameter of 66. We then send the following to the sensor:

0xf9 0x42 0x42

We receive the following response from the sensor:

0x17 0x39 0x15 0x93 0x0c 0xc4 0x86 0x0 0x0 0xc5 0x54 0x0 0x0 0x46 0x7c 0xc0 0x0

Going in order, we used bits 1 and 6, so we can parse out the timestamp first, which is 4 bytes, and then the data length, which is 1 byte:

Timestamp: 0x17 0x39 0x15 0x93 (389617043)

Data Length: 0x0c (12)

Data: 0xc4 0x86 0x0 0x0 0xc5 0x54 0x0 0x0 0x46 0x7c 0xc0 0x0 (-1072.0, -3392.0, 16176.0)

For the ascii version, we would send the following:

“;66\n”

We would receive the following response:

“389617043,37,-1072.00000,-3392.00000,16176.00000\r\n”

4.3.2 Wired Streaming with Response Header

Streaming data can also have Response Header data prepended to each streamed packet. This can be accomplished by calling the Start Streaming command (0x55) with the Response Header Packet Byte. Assuming that streaming has been configured properly and a non-zero Wired Response Header bitfield has been set, the following examples will start streaming with Response Headers disabled and enabled, respectively:

```
0xf7 0x55 0x55          //Start streaming WITHOUT response header prepended
0xf9 0x55 0x55          //Start streaming WITH response header prepended
```

Keep in mind that the actual start command will also have a Response Header attached that must be successfully parsed.

4.4 Command Overview

There are over 90 different command messages that are grouped numerically by function. Unused command message bytes are reserved for future expansion.

When looking at the following command message tables, note the following:

- The "Data Len" field indicates the number of additional data-bytes the command expects to follow the command-byte itself. This number doesn't include the Start of Packet, Command, or Checksum bytes. Thus, the total message size can be calculated by adding three bytes to the "Data Len" listed in the table.
- Likewise, the "Return Data Len" field indicates the number of data-bytes the command delivers back to the sender once the command has finished executing.
- Under "Return Data Details", each command lists the sort of data which is being returned and next to this in parenthesis the form this data takes. For example, a quaternion is represented by 4 floating point numbers, so a command which returns a quaternion would list "Quaternion(float x4)" for its return data details.
- Command length information only applies to binary commands, as ascii commands can vary in length.
- For quaternions, data is always returned in x, y, z, w order.
- Euler angles are always returned in pitch, yaw, roll order.
- When calling commands in ASCII mode, there is no fixed byte length for the parameter data or return data, as the length depends on the ASCII encoding.

4.4.1 Orientation Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
0(0x00)	Get tared orientation as quaternion	Returns the filtered, tared orientation estimate in quaternion form	16	Quaternion (float x4)	0	
1(0x01)	Get tared orientation as euler angles	Returns the filtered, tared orientation estimate in euler angle form	12	Euler Angles (float x3)	0	
2(0x02)	Get tared orientation as rotation matrix	Returns the filtered, tared orientation estimate in rotation matrix form	36	Rotation Matrix (float x9)	0	
3(0x03)	Get tared orientation as axis angle	Returns the filtered, tared orientation estimate in axis-angle form	16	Axis (float x3), Angle in Radians (float)	0	
4 (0x04	Get tared orientation as two vector.	Returns the filtered, tared orientation estimate in two vector form, where the first vector refers to forward and the second refers to down.	24	Forward Vector (float x3), Down Vector (float x3)	0	
5(0x05)	Get difference quaternion	Returns the difference between the measured orientation from last frame and this frame.	16	Quaternion (float x4)	0	
6(0x06)	Get untared orientation as quaternion	Returns the filtered, untared orientation estimate in quaternion form.	16	Quaternion (float x4)	0	
7(0x07)	Get untared orientation as euler angles	Returns the filtered, untared orientation estimate in euler angle form	12	Euler Angles (float x3)	0	
8(0x08)	Get untared orientation as rotation matrix	Returns the filtered, untared orientation estimate in rotation matrix form	36	Rotation Matrix (float x9)	0	
9(0x09)	Get untared orientation as axis angle	Returns the filtered, untared orientation estimate in axis-angle form	16	Axis (float x3), Angle in Radians (float)	0	
10(0x0A)	Get untared orientation as two vector.	Returns the filtered, untared orientation estimate in two vector form, where the first vector refers to north and the second refers to gravity.	24	North Vector (float x3), Gravity Vector (float x3)	0	
11(0x0B)	Get tared two vector in sensor frame	Returns the filtered, tared orientation estimate in two vector form, where the first vector refers to forward and the second refers to down. These vectors are given in the sensor reference frame and not the global reference frame.	24	Forward Vector (float x3), Down Vector (float x3)	0	
12(0x0C)	Get untared two vector in sensor frame	Returns the filtered, untared orientation estimate in two vector form, where the first vector refers to forward and the second refers to down. These vectors are given in the sensor reference frame and not the global reference frame.	24	North Vector (float x3), Gravity Vector (float x3)	0	

4.4.2 Normalized Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
32(0x20)	Get all normalized component sensor data	Returns the normalized gyro rate vector, accelerometer vector, and compass vector. Note that the gyro vector is in units of radians/sec, while the accelerometer and compass are unit-length vectors indicating the direction of gravity and north, respectively. These two vectors do not have any magnitude data associated with them.	36	Gyro Rate in units of radians/sec (Vector x3), Gravity Direction (Vector x3), North Direction (Vector x3)	0	
33(0x21)	Get normalized gyro rate	Returns the normalized gyro rate vector, which is in units of radians/sec.	12	Gyro Rate in units of radians/sec (float x3)	0	
34(0x22)	Get normalized accelerometer vector	Returns the normalized accelerometer vector. Note that this is a unit-vector indicating the direction of gravity. This vector does not have any magnitude data associated with it.	12	Gravity Direction (Vector x3)	0	
35(0x23)	Get normalized compass vector	Returns the normalized compass vector. Note that this is a unit-vector indicating the direction of gravity. This vector does not have any magnitude data associated with it.	12	North Direction (Vector x3)	0	

4.4.3 Corrected Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
37(0x25)	Get all corrected component sensor data	Returns the corrected gyro rate vector, accelerometer vector, and compass vector. Note that the gyro vector is in units of radians/sec, the accelerometer vector is in units of G, and the compass vector is in units of gauss.	36	Gyro Rate in units of radians/sec (Vector x3), Acceleration Vector in units of G (Vector x3), Compass Vector in units of gauss (Vector x3)	0	
38(0x26)	Get corrected gyro rate	Returns the corrected gyro rate vector, which is in units of radians/sec. Note that this result is the same data returned by the normalized gyro rate command.	12	Gyro Rate in units of radians/sec (float x3)	0	
39(0x27)	Get corrected accelerometer vector	Returns the acceleration vector in units of G. Note that this acceleration will include the static component of acceleration due to gravity.	12	Acceleration Vector in units of G (float x3)	0	
40(0x28)	Get corrected compass vector	Returns the compass vector in units of gauss.	12	Compass Vector in units of gauss (float x3)	0	
41(0x29)	Get corrected linear acceleration in global space	Returns the linear acceleration of the device, which is the overall acceleration which has been orientation compensated and had the component of acceleration due to gravity removed. Uses the tared orientation.	12	Acceleration Vector in units of G (float x3)	0	
48(0x30)	Correct raw gyro data	Converts the supplied raw data gyroscope vector to its corrected data representation.	12	Gyro Rate in units of radians/sec (float x3)	12	Gyro Rate in counts per degrees/sec (Vector x3)
49(0x31)	Correct raw accel data	Converts the supplied raw data accelerometer vector to its corrected data representation.	12	Acceleration Vector in units of G (float x3)	12	Acceleration Vector in counts per g (Vector x3)
50(0x32)	Correct raw compass data	Converts the supplied raw data compass vector to its corrected data representation.	12	Compass Vector in units of gauss (float x3)	12	Compass Vector in counts per gauss (Vector x3)

4.4.4 Other Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
43(0x2B)	Get temperature C	Returns the temperature of the sensor in Celsius.	4	Temperature (float)	0	
44(0x2C)	Get temperature F	Returns the temperature of the sensor in Fahrenheit	4	Temperature (float)	0	
45(0x2D)	Get confidence factor	Returns a value indicating how much the sensor is being moved at the moment. This value will return 1 if the sensor is completely stationary, and will return 0 if it is in motion. This command can also return values in between indicating how much motion the sensor is experiencing.	4	Confidence Factor (float)	0	

4.4.5 Raw Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
64(0x40)	Get all raw component sensor data	Returns the raw gyro rate vector, accelerometer vector and compass vector as read directly from the component sensors without any additional post-processing. The range of values is dependent on the currently selected range for each respective sensor.	36	Gyro Rate in counts per degrees/sec (Vector x3), Acceleration Vector in counts per g (Vector x3), Compass Vector in counts per gauss (Vector x3)	0	
65(0x41)	Get raw gyroscope rate	Returns the raw gyro rate vector as read directly from the gyroscope without any additional post-processing.	12	Gyro Rate in counts per degrees/sec (Vector x3)	0	
66(0x42)	Get raw accelerometer data	Returns the raw acceleration vector as read directly from the accelerometer without any additional post-processing.	12	Acceleration Vector in counts per g (Vector x3)	0	
67(0x43)	Get raw compass data	Returns the raw compass vector as read directly from the compass without any additional post-processing.	12	Compass Vector in counts per gauss (Vector x3)	0	

4.4.6 Data-Logging Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
59(0x3B)	Format and Initialize SD Card	Erases the contents of the SD card.	0		0	
60(0x3C)	Begin data logging session	Initiates a data logging section with the specified attributes as indicated in the provided data logging configuration file.	0		0	
61(0x3D)	End data logging session	Terminates the ongoing data logging session	0		0	
62(0x3E)	Set clock values	Sets the current time on the onboard real-time clock.			6	Month (Byte), Day (Byte), Year (Byte), Hour (Byte), Minute (Byte), Second (Byte)
63(0x3F)	Get clock values	Returns the current time as read by the onboard real-time clock.	6	Month (Byte), Day (Byte), Year (Byte), Hour (Byte), Minute (Byte), Second (Byte)	0	

4.4.7 Streaming Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
80(0x50)	Set streaming slots	Configures data output slots for streaming mode. Command accepts a list of eight bytes, where each byte corresponds to a different data command. Every streaming iteration, each command will be executed in order and the resulting data will be output in the specified slot. Valid commands are commands in the ranges 0x0 – 0x10, 0x20 – 0x30, 0x40 – 0x50, 0xC9 – 0xCA (for battery-powered sensors) and 0xFA. A slot value of 0xFF 'clears' the slot and prevents any data from being written in that position. This command can fail if there is an invalid command passed in as any of the parameters or if the total allotted size is exceeded. Upon failure, all slots will be reset to 0xFF. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		8	Commands (Byte x8)
81(0x51)	Get streaming slots	Returns the current streaming slots configuration.	8	Commands (Byte x8)	0	
82(0x52)	Set streaming timing	Configures timing information for a streaming session. All parameters are specified in microseconds. The first parameter is the interval, which specifies how often data will be output. A value of 0 means that data will be output at the end of every filter loop. Aside from 0, values lower than 1000 will be clamped to 1000. The second parameter is the duration, which specifies the length of the streaming session. If this value is set to 0xFFFFFFFF, streaming will continue indefinitely until it is stopped via command 0x56. The third parameter is the delay, which specifies a n amount of time the sensor will wait before outputting the first packet of streaming data. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		12	Interval (Unsigned int), Duration (Unsigned int), Delay (Unsigned int)
83(0x53)	Get streaming timing	Returns the current streaming timing information.	12	Interval (Unsigned int), Duration (Unsigned int), Delay (Unsigned int)	0	
84(0x54)	Get streaming batch	Return a single packet of streaming data using the current slot configuration.	Varies		0	
85(0x55)	Start streaming	Start a streaming session using the current slot and timing configuration.	0		0	
86(0x56)	Stop streaming	Stop the current streaming session.	0		0	
95(0x5F)	Update current timestamp	Set the current internal timestamp to the specified value.	0		4	Timestamp (Unsigned int)

4.4.8 Configuration Write Commands

16(0x10)	Set euler angle decomposition order	Sets the current euler angle decomposition order, which determines how the angles returned from command 0x1 are decomposed from the full quaternion orientation. Possible values are 0x0 for XYZ, 0x1 for YZX, 0x2 for ZXY, 0x3 for ZYX, 0x4 for XZY or 0x5 for YXZ (default).	0		1	Euler angle decomposition order (byte)
17(0x11)	Set magnetoresistive threshold	Sets required parameters that are necessary to trigger magnetometer resistance mode. First parameter to the command specifies the change in magnetometer field strength that is required to trigger the resistance. Once this field has been detected, the magnetometer will enter a period where it is completely locked out of the orientation calculation—this period will increase while magnetic perturbations are still being detected, but will dissipate as the sensor remains stationary. Once this period is over, the sensor orientation will slowly begin trusting the magnetometer again. The second parameter represents the number of frames that must elapse before the magnetometer is fully trusted again. The third parameter represents a decay value between 0 and 1 that indicates how quickly the outright magnetometer rejection state will fall off. Values closer to 1 result in the magnetometer rejection lasting longer. The final parameter represents how quickly a magnetic perturbation is detected. Values closer to 1 result in the magnetometer rejection occurring more slowly. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		16	Magnetoresistive threshold in gauss(float), Number of magnetometer trust frames (unsigned int), magnetometer lockout decay value (float), magnetometer perturbation detection value (float)
18(0x12)	Set accelerometer resistance threshold	Sets required parameters that are necessary to trigger accelerometer rejection. During the accelerometer rejection period, the contribution of the accelerometer to the selected orientation estimation algorithm will be zero. The arguments to this command specify the accelerometer threshold and the number of frames that the rejection is active, respectively. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		8	Accelerometer threshold in g's (float), Number of accelerometer lockout frames (unsigned int),
19(0x13)	Offset with current orientation	Sets the offset orientation to be the same as the current filtered orientation.	0		0	
20(0x14)	Reset base offset	Sets the base offset to an identity quaternion.	0		0	
21(0x15)	Offset with quaternion	Sets the offset orientation to be the same as the supplied orientation, which should be passed as a quaternion.	0		16	Quaternion (float x4)
22(0x16)	Set base offset with current orientation	Sets the base offset orientation to be the same as the current filtered orientation.	0		0	
96(0x60)	Tare with current orientation	Sets the tare orientation to be the same as the current filtered orientation.	0		0	
97(0x61)	Tare with quaternion	Sets the tare orientation to be the same as the supplied orientation, which should be passed as a quaternion.	0		16	Quaternion (float x4)
98(0x62)	Tare with rotation matrix	Sets the tare orientation to be the same as the supplied orientation, which should be passed as a rotation matrix.	0		36	Rotation Matrix (float x9)
99(0x63)	Set static accelerometer trust value	Determines how trusted the accelerometer contribution is to the overall orientation estimation. Trust is 0 to 1, with 1 being fully trusted and 0 being not trusted at all.	0		4	Accelerometer trust value (float)
100(0x64)	Set confidence accelerometer trust values	Determines how trusted the accelerometer contribution is to the overall orientation estimation. Instead of using a single value, uses a minimum and maximum value. Trust values will be selected from this range depending on the confidence factor. This can have the effect of smoothing out the accelerometer when the sensor is in motion.	0		8	Minimum accelerometer trust value (float), Maximum accelerometer trust value (float)
101(0x65)	Set static compass trust value	Determines how trusted the accelerometer contribution is to the overall orientation estimation. tribution is to the overall orientation estimation. Trust is 0 to 1, with 1 being fully trusted and 0 being not trusted at all.	0		4	Compass trust value (float)

102(0x66)	Set confidence compass trust values	Determines how trusted the compass contribution is to the overall orientation estimation. Instead of using a single value, uses a minimum and maximum value. Trust values will be selected from this range depending on the confidence factor. This can have the effect of smoothing out the compass when the sensor is in motion.	0		8	Minimum compass trust value (float), Maximum compass trust value (float)
103(0x67)	Set desired update rate	Causes the processor to wait for the specified number of microseconds at the end of each update loop. Can be useful for bounding the overall update rate of the sensor if necessary.	0		4	Microsecond update rate (unsigned integer)
104(0x68)	Set multi reference vectors with current orientation	Uses the current tared orientation to set up the reference vector for the nearest orthogonal orientation. This is an advanced command that is best used through 3-Space Sensor Suite calibration utilities. For more information, please refer to the 3-Space Sensor Suite Quick Start Guide.	0		0	
105(0x69)	Set reference vector mode	Set the current reference vector mode. Parameter can be 0 for single static mode, which uses a certain reference vector for the compass and another certain vector for the accelerometer at all times, 1 for single auto mode, which uses (0, -1, 0) as the reference vector for the accelerometer at all times and uses the average angle between the accelerometer and compass to calculate the compass reference vector once upon initiation of this mode, 2 for single auto continuous mode, which works similarly to single auto mode, but calculates this continuously, or 3 for multi-reference mode, which uses a collection of reference vectors for the compass and accelerometer both, and selects which ones to use before each step of the filter.	0		1	Mode (Byte)
106(0x6A)	Set oversample rate	Sets the number of times to sample each component sensor for each iteration of the filter. This can smooth out readings at the cost of responsiveness. If this value is set to 0 or 1, no oversampling occurs—otherwise, the number of samples per iteration depends on the specified parameter, up to a maximum of 65535. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		6	Gyro Samples (unsigned short), Accel Samples (unsigned short), Compass Samples (unsigned short)
107(0x6B)	Set gyroscope enabled	Enable or disable gyroscope readings as inputs to the orientation estimation. Note that updated gyroscope readings are still accessible via commands. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Enabled (Byte)
108(0x6C)	Set accelerometer enabled	Enable or disable accelerometer readings as inputs to the orientation estimation. Note that updated accelerometer readings are still accessible via commands. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Enabled (Byte)
109(0x6D)	Set compass enabled	Enable or disable compass readings as inputs to the orientation estimation. Note that updated compass readings are still accessible via commands. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Enabled (Byte)
110(0x6E)	Reset multi-reference vectors to zero	Resets all reference vectors in the multi-reference table to zero. Intended for advanced users.	0		0	
111(0x6F)	Set multi-reference table resolution	Sets the number of cell dimensions and number of nearby vectors per cell for the multi-reference lookup table. First parameter indicates the number of cell divisions—as an example, multi-reference mode, by default, only handles orientations reachable by successive rotations of ninety degrees about any of the three axes, and hence, has a resolution of 4 (360 / 4 == 90). Thus, a resolution of 8 would provide rotations of forty-five degrees about any of the three axes (360 / 8 == 45). The second parameter indicates the number of adjacent vectors that will be checked for each In addition, the number of checked vectors can be adjusted as well. The second parameters refers to the number of adjacent reference vectors that are 'averaged' to produce the final reference vector for the particular orientation, up to a maximum of 32. Intended for advanced users.	0		2	Resolution (Byte), Number of Check Vectors (Byte)

112(0x70)	Set compass mulfi-reference vector	Directly set the multi-reference compass vector at the specified index. First parameter is index, second parameter is compass vector. Intended for advanced users.	0		13	Index (Byte), Compass Reference Vector (float x3)
113(0x71)	Set compass multi-reference check vector	Set the compass reading to be used as a check vector to determine which cell index to draw the reference vector from. First parameter is an index, second parameter is the compass vector. Intended for advanced users.	0		13	Index (Byte), Compass Check Vector (float x3)
114(0x72)	Set accelerometer multi-reference vector	Directly set the multi-reference accelerometer vector at the specified index. First parameter is index, second parameter is compass vector. Intended for advanced users.	0		13	Index (Byte), Accelerometer Reference Vector (float x3)
115(0x73)	Set accelerometer multi-reference check vector	Set the accelerometer reading to be used as a check vector to determine which cell index to draw the reference vector from. First parameter is an index, second parameter is the accelerometer vector. Intended for advanced users.	0		13	Index (Byte), Accelerometer Check Vector (float x3)
116(0x74)	Set axis directions	Sets alternate directions for each of the natural axes of the sensor. The only parameter is a bitfield representing the possible combinations of axis swapping. The lower 3 bits specify where each of the natural axes appears: 000: X: Right, Y: Up, Z: Forward (left-handed system, standard operation) 001: X: Right, Y: Forward, Z: Up (right-handed system) 002: X: Up, Y: Right, Z: Forward (right-handed system) 003: X: Forward, Y: Right, Z: Up (left-handed system) 004: X: Up, Y: Forward, Z: Right (left-handed system) 005: X: Forward, Y: Up, Z: Right (right-handed system) (For example, using X: Right, Y: Forward, Z: Up means that any values that appear on the positive vertical(Up) axis of the sensor will be the third(Z) component of any vectors and will have a positive sign, and any that appear on the negative vertical axis will be the Z component and will have a negative sign.) The 3 bits above those are used to indicate which axes, if any, should be reversed. If it is cleared, the axis will be pointing in the positive direction. Otherwise, the axis will be pointed in the negative direction. (Note: These are applied to the axes after the previous conversion takes place). Bit 4: Positive/Negative Z (Third resulting component) Bit 5: Positive/Negative Y (Second resulting component) Bit 6: Positive/Negative X (First resulting component) Note that for each negation that is applied, the handedness of the system flips. So, if X and Z are negative and you are using a left-handed system, the system will still be left handed, but if only X is negated, the system will become right-handed.	0		1	Axis Direction Byte (byte)
117(0x75)	Set running average percent	Sets what percentage of running average to use on a component sensor, or on the sensor's orientation. This is computed as follows: total_value = total_value* percent total_value = total_value + current_value * (1 – percent) current_value = total_value If the percentage is 0, the running average will be shut off completely. Maximum value is 1. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		16	Gyro percent (float), accel percent (float), compass percent (float), orientation percent (float)

118(0x76)	Set compass reference vector	Sets the static compass reference vector for Single Reference Mode.	0		12	Compass Reference Vector (float x3)
119(0x77)	Set accelerometer reference vector	Sets the static accelerometer reference vector for Single Reference Mode.	0		12	Accelerometer Reference Vector (float x3)
120(0x78)	Reset filter	Resets the state of the currently selected filter	0		0	
121(0x79)	Set accelerometer range	Only parameter is the new accelerometer range, which can be 0 for ±2g (Default range), which can be 1 for ±4g, or 2 for ±8g. Higher ranges can detect and report larger accelerations, but are not as accurate for smaller accelerations. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Accelerometer range setting (byte)
122(0x7a)	Set multi-reference weight power	Set weighting power for multi reference vector weights. Multi reference vector weights are all raised to the weight power before they are summed and used in the calculation for the final reference vector. Setting this value nearer to 0 will cause the reference vectors to overlap more, and setting it nearer to infinity will cause the reference vectors to influence a smaller set of orientations.	0		4	Weight power (float)
123(0x7b)	Set filter mode	Used to disable the orientation filter or set the orientation filter mode. Changing this parameter can be useful for tuning filter-performance versus orientation-update rates. Passing in a parameter of 0 places the sensor into IMU mode, a 1 places the sensor into Kalman Filtered Mode (Default mode), a 2 places the sensor into Alternating Kalman Filter Mode, a 3 places the sensor into Complementary Filter Mode, a 4 places the sensor into Quaternion Gradient Descent Filter Mode, and a 5 places the sensor into Magnetoresistive Quaternion Gradient Descent Filter Mode. More information can be found in Section 3.1.5. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Mode (Byte)
124(0x7c)	Set running average mode	Used to further smooth out the orientation at the cost of higher latency. Passing in a parameter of 0 places the sensor into a static running average mode, a 1 places the sensor into a confidence-based running average mode, which changes the running average factor based upon the confidence factor, which is a measure of how 'in motion' the sensor is. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Mode (Byte)
125(0x7d)	Set gyroscope range	Only parameter is the new gyroscope range, which can be 0 for ±250 DPS, 1 for ±500 DPS, or 2 for ±2000 DPS (Default range). Higher ranges can detect and report larger angular rates, but are not as accurate for smaller angular rates. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Gyroscope range setting (Byte)
126(0x7e)	Set compass range	Only parameter is the new compass range, which can be 0 for ±0.88G, 1 for ±1.3G (Default range), 2 for ±1.9G, 3 for ±2.5G, 4 for ±4.0G, 5 for ±4.7G, 6 for ±5.6G, or 7 for ±8.1G. Higher ranges can detect and report larger magnetic field strengths but are not as accurate for smaller magnetic field strengths. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Compass range setting (Byte)

4.4.9 Configuration Read Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
128(0x80)	Get tare orientation as quaternion	Returns the current tare orientation as a quaternion.	16	Quaternion (float x4)	0	
129(0x81)	Get tare orientation as rotation matrix	Returns the current tare orientation as a rotation matrix.	36	Rotation Matrix (float x9)	0	
130(0x82)	Get accelerometer trust values	Returns the current accelerometer min and max trust values. If static trust values were set, both of these will be the same.	8	Accelerometer trust values, min and max (float x2)	0	
131(0x83)	Get compass trust values	Returns the current compass min and max trust values. If static trust values were set, both of these will be the same.	8	Compass trust values, min and max (float x2)	0	
132(0x84)	Get current update rate	Reads the amount of time taken by the last filter update step.	4	Last update time in microseconds (int)	0	
133(0x85)	Get compass reference vector	Reads the current compass reference vector. Note that this is not valid if the sensor is in Multi Reference Vector mode.	12	Compass reference vector (float x3)	0	
134(0x86)	Get accelerometer reference vector	Reads the current compass reference vector. Note that this is not valid if the sensor is in Multi Reference Vector mode.	12	Accelerometer reference vector (float x4)	0	
135(0x87)	Get reference vector mode	Reads the current reference vector mode. Return value can be 0 for single static, 1 for single auto, 2 for single auto continuous or 3 for multi.	1	Mode (byte)	0	
136(0x88)	Get compass multi-reference vector	Reads the multi-reference mode compass reference vector at the specified index. Intended for advanced users.	12	Compass multi-reference reference vector (float x3)	1	Index (byte)
137(0x89)	Get compass multi-reference check vector	Reads the multi-reference mode compass reference check vector at the specified index. Intended for advanced users.	12	Compass multi-reference reference check vector (float x3)	1	Index (byte)
138(0x8a)	Get accelerometer multi-reference vector	Reads the multi-reference mode accelerometer reference vector at the specified index. Intended for advanced users.	12	Accelerometer multi-reference reference vector (float x3)	1	Index (byte)
139(0x8b)	Get accelerometer multi-reference check vector	Reads the multi-reference mode accelerometer reference check vector at the specified index. Intended for advanced users.	12	Accelerometer multi-reference reference check vector (float x3)	1	Index (byte)
140(0x8c)	Get gyroscope enabled state	Returns a value indicating whether the gyroscope contribution is currently part of the orientation estimate: 0 for off, 1 for on.	1	Gyroscope enabled value (byte)	0	
141(0x8d)	Get accelerometer enabled state	Returns a value indicating whether the accelerometer contribution is currently part of the orientation estimate: 0 for off, 1 for on.	1	Accelerometer enabled value (byte)	0	
142(0x8e)	Get compass enabled state	Returns a value indicating whether the compass contribution is currently part of the orientation estimate: 0 for off, 1 for on.	1	Compass enabled value (byte)	0	
143(0x8f)	Get axis direction	Returns a value indicating the current axis direction setup. For more information on the meaning of this value, please refer to the Set Axis Direction command (116).	1	Axis direction value (byte)	0	
144(0x90)	Get oversample rate	Returns values indicating how many times each component sensor is sampled before being stored as raw data. A value of 1 indicates that no oversampling is taking place, while a value that is higher indicates the number of samples per component sensor per filter update step.	6	Gyro Samples (unsigned short), Accel Samples (unsigned short), Compass Samples (unsigned short)	0	
145(0x91)	Get running average percent	Returns the running average percent value for each component sensor and for the orientation. The value indicates what portion of the previous reading is kept and incorporated into the new reading.	16	Gyro percent (float), accel percent (float), compass percent (float), orientation percent (float)	0	
146(0x92)	Get desired update rate	Returns the current desired update rate. Note that this value does not indicate the actual update rate, but instead indicates the value that should be spent 'idling' in the main loop. Thus, without having set a specified desired update rate, this value should read 0.	4	Desired update rate in microseconds (int)	0	
148(0x94)	Get accelerometer range	Return the current accelerometer measurement range, which can be a 0 for ±2g, 1 for ±4g or a 2 for ±8g.	1	Accelerometer range setting (byte)	0	
149(0x95)	Get multi-reference mode power weight	Read weighting power for multi-reference vector weights. Intended for advanced users.	4	Weight (float)	0	

150(0x96)	Get multi-reference resolution	Reads number of cell divisions and number of nearby vectors per cell for the multi-reference vector lookup table. For more information on these values, please refer to the Set Multi-Reference Resolution command (111). Intended for advanced users.	2	Number of cell divisions (byte), number of nearby vectors (byte)	0	
151(0x97)	Get number of multi-reference cells	Reads the total number of multi-reference cells. Intended for advanced users.	4	Number of cells (int)	0	
152(0x98)	Get filter mode	Returns the current filter mode, which can be 0 for IMU mode, 1 for Kalman, 2 for Alternating Kalman, 3 for Complementary, or 4 for Quaternion Gradient Descent. For more information, please refer to the Set Filter Mode command (123).	1	Filter mode (byte)	0	
153(0x99)	Get running average mode	Reads the selected mode for the running average, which can be 0 for normal or 1 for confidence.	1	Running average mode (byte)	0	
154(0x9a)	Get gyroscope range	Reads the current gyroscope measurement range, which can be 0 for ±250 DPS, 1 for ±500 DPS or 2 for ±2000 DPS.	1	Gyroscope range setting (byte)	0	
155(0x9b)	Get compass range	Reads the current compass measurement range, which can be 0 for ±0.88G, 1 for ±1.3G, 2 for ±1.9G, 3 for ±2.5G, 4 for ±4.0G, 5 for ±4.7G, 6 for ±5.6G or 7 for ±8.1G.	1	Compass range setting (byte)	0	
156(0x9c)	Get euler angle decomposition order	Reads the current euler angle decomposition order.	1	Euler angle decomposition order (byte)	0	
157(0x9d)	Get magnetoresistive threshold	Reads the current magnetoresistive threshold parameters.	16	Magnetoresistive threshold in gauss(float), Number of magnetometer trust frames (unsigned int), magnetometer lockout decay value (float), magnetometer perturbation detection value (float)	0	
158(0x9e)	Get accelerometer resistance threshold	Reads the current accelerometer threshold parameters.	8	Accelerometer threshold in g's (float), Number of accelerometer lockout frames (unsigned int),	0	
159(0x9f)	Get offset orientation as quaternion	Returns the current offset orientation as a quaternion.	16	Quaternion (float x4)	0	

4.4.10 Calibration Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
160(0xa0)	Set compass calibration coefficients	Sets the current compass calibration parameters to the specified values. These consist of a bias which is added to the raw data vector and a matrix by which the value is multiplied. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		48	Matrix (float x9), Bias (float x3)
161(0xa1)	Set accelerometer calibration coefficients	Sets the current accelerometer calibration parameters to the specified values. These consist of a bias which is added to the raw data vector and a matrix by which the value is multiplied. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		48	Matrix (float x9), Bias (float x3)
162(0xa2)	Get compass calibration coefficients	Return the current compass calibration parameters.	48	Matrix (float x9), Bias (float x3)		
163(0xa3)	Get accelerometer calibration coefficients	Return the current accelerometer calibration parameters.	48	Matrix (float x9), Bias (float x3)		
164(0xa4)	Get gyroscope calibration coefficients	Return the current gyroscope calibration parameters.	48	Matrix (float x9), Bias (float x3)		
165(0xa5)	Begin gyroscope auto-calibration	Performs auto-gyroscope calibration. Sensor should remain still while samples are taken. The gyroscope bias will be automatically placed into the bias part of the gyroscope calibration coefficient list.	0		0	
166(0xa6)	Set gyroscope calibration coefficients	Sets the current gyroscope calibration parameters to the specified values. These consist of a bias which is added to the raw data vector and a matrix by which the value is multiplied. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		48	Matrix (float x9), Bias (float x3)
169(0xa9)	Set calibration mode	Bias, 1 for Scale-Bias and 2 for Ortho-Calibration. For more information, refer to section 3.1.3 Additional Calibration. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Mode (Byte)
170(0xaa)	Get calibration mode	Reads the current calibration mode, which can be 0 for Bias, 1 for Scale-Bias or 2 for Ortho-Calibration. For more information, refer to section 3.1.3 Additional Calibration.	1	Mode (byte)	0	
171(0xab)	Set ortho-calibration data point from current orientation	Set the ortho-calibration compass and accelerometer vectors corresponding to this orthogonal orientation. Intended for advanced users.	0		0	
172(0xac)	Set ortho-calibration data point from vector	Directly set a vector corresponding to this orthogonal orientation. First parameter is type, where 0 is for compass and 1 is for accelerometer. Second parameter is index, which indicates the orthogonal orientation. Intended for advanced users.	0		14	Type (Byte), Index (Byte), Accelerometer or Compass Vector (float x3)
173(0xad)	Get ortho-calibration data point	Return the vector corresponding to the orthogonal orientation given by index. First parameter is type, where 0 is for compass and 1 is for accelerometer. Second parameter is index, which indicates the orthogonal orientation. Intended for advanced users.	12	Accelerometer or compass vector (float x3)	2	Type (Byte), Index (Byte)
174(0xae)	Perform ortho-calibration	Stores accelerometer and compass data in the ortho-lookup table for use in the orientation fusion algorithm. For best results, each of the 24 orientations should be filled in with component sensor data. Note also that ortho-calibration data will not be used unless the calibration mode is set to Ortho-Calibration. For more information, refer to Section 3.1.3 Additional Calibration. Intended for advanced users.	0		0	
175(0xaf)	Clear ortho-calibration data	Clear out all ortho-lookup table data. Intended for advanced users.	0		0	

4.4.11 Battery Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
201(0xc9)	Get battery voltage	Read the current battery level in volts. Note that this value will read as slightly higher than it actually is if it is read via a USB connection.	4	Battery level in voltage (float)	0	
202(0xca)	Get battery percent remaining	Read the current battery lifetime as a percentage of the total. Note that this value will read as slightly higher than it actually is if it is read via a USB connection.	1	Battery level as percent (byte)	0	
203(0xcb)	Get battery status	Returns a value indicating the current status of the battery, which can be a 3 to indicate that the battery is currently not charging, a 2 to indicate that the battery is charging and thus plugged in, or a 1 to indicate that the sensor is fully charged.	1	Battery charge status (byte)	0	

4.4.12 General Commands

196(0xc4)	Set LED Mode	Allows finer-grained control over the sensor LED. Accepts a single parameter that can be 0 for standard, which displays all standard LED status indicators or 1 for static, which displays only the LED color as specified by command 238.	0		1	LED mode (byte)
200(0xc8)	Get LED Mode	Returns the current sensor LED mode, which can be 0 for standard or 1 for static.	1	LED mode (byte)	0	
221(0xdd)	Set wired response header bitfield	Configures the response header for data returned over a wired connection. The only parameter is a four-byte bitfield that determines which data is prepended to all data responses. The following bits are used: 0x1: (1 byte) Success/Failure, with non-zero values representing failure. 0x2: (4 bytes) Timestamp, in microseconds. 0x4: (1 byte) Command echo—outputs the called command. Returns 0xFF for streamed data. 0x8: (1 byte) Additive checksum over returned data, but not including response header. 0x10: (1 byte) Logical ID, returns 0xFE for wired sensors. Meant to be used with 3-Space Dongle response header (For more info, see command 0xDB). 0x20: (4 bytes) Serial number 0x40: (1 byte) Data length, returns the length of the requested data, not including response header. This setting can be committed to non-volatile flash memory by calling the Commit Settings command. For more information on Response Headers, please refer to Section 4.4.	0		4	Response header configuration (Unsigned int)
222(0xde)	Get wired response header bitfield	Return the current wired response header bitfield. For more information, please refer to Section 4.4.	4	Response header configuration (Unsigned int)	0	
223(0xdf)	Get firmware version string	Returns a string indicating the current firmware version.	12	Firmware version (string)	0	
224(0xe0)	Restore factory settings	Return all non-volatile flash settings to their original, default settings.	0		0	
225(0xe1)	Commit settings	Commits all current sensor settings to non-volatile flash memory, which will persist after the sensor is powered off. For more information on which parameters can be stored in this manner, refer to Section 3.4 Sensor Settings.	0		0	
226(0xe2)	Software reset	Resets the sensor.	0		0	
227(0xe3)	Set sleep mode	Sets the current sleep mode of the sensor. Supported sleep modes are 0 for NONE and 1 for IDLE. IDLE mode merely skips all filtering steps. NONE is the default state.	0		1	Sleep mode (byte)
228(0xe4)	Get sleep mode	Reads the current sleep mode of the sensor, which can be 0 for NONE or 1 for IDLE.	1	Sleep mode (byte)	0	
229(0xe5)	Enter bootloader mode	Places the sensor into a special mode that allows firmware upgrades. This will case normal operation until the firmware update mode is instructed to return the sensor to normal operation. For more information on upgrading firmware, refer to the 3-Space Sensor Suite Quick Start Guide.	0		0	
230(0xe6)	Get hardware version string	Returns a string indicating the current hardware version.	32	Hardware version (string)	0	
231(0xe7)	Set UART baud rate	Sets the baud rate of the physical UART. This setting does not need to be committed, but will not take effect until the sensor is reset. Valid baud rates are 1200, 2400, 4800, 9600, 19200, 28800, 38400, 57600, 115200 (default), 230400, 460800 and 921600. Note that this is only applicable for sensor types that have UART interfaces.	0		4	Baud rate (int)
232(0xe8)	Get UART baud rate	Returns the baud rate of the physical UART. Note that this is only applicable for sensor types that have UART interfaces.	4	Baud rate (int)	0	

233(0xe9)	Set USB Mode	Sets the communication mode for USB. Accepts one value that can be 0 for CDC (default) or 1 for FTDI.	0		1	USB communication mode (byte)
234(0xea)	Get USB Mode	Returns the current USB communication mode.	1	USB communication mode (byte)	0	
237(0xed)	Get serial number	Returns the serial number, which will match the value etched onto the physical sensor.	4	Serial number (int)	0	
238(0xee)	Set LED color	Sets the color of the LED on the sensor to the specified RGB color. This setting can be committed to non-volatile flash memory by calling the Commit Wireless Settings command.	0		12	RGB Color (float x3)
239(0xef)	Get LED color	Returns the color of the LED on the sensor.	12	RGB Color (float x3)	0	

4.4.13 Wired HID Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
240(0xf0)	Enable/disable joystick	Enable or disable streaming of joystick HID data for this sensor.	0		1	Joystick enabled state (byte)
241(0xf1)	Enable/disable mouse	Enable or disable streaming of mouse HID data for this sensor.	0		1	Mouse enabled state (byte)
242(0xf2)	Get joystick enabled	Read whether the sensor is currently streaming joystick HID data.	1	Joystick enabled state (byte)	0	
243(0xf3)	Get mouse enabled	Read whether the sensor is currently streaming mouse HID data.	1	Mouse enabled state (byte)	0	

4.4.14 General HID Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
244(0xf4)	Set control mode	Sets the operation mode for one of the controls. The first parameter is the control class,which can be 0 for Joystick Axis, 1 for Joystick Button, 2 for Mouse Axis or 3 for Mouse Button. There are two axes and eight buttons on the joystick and mouse. The second parameter, the control index, selects which one of these axes or buttons you would like to modify. The third parameter, the handler index, specifies which handler you want to take care of this control. These can be the following: Turn off this control: 255 Axes: Global Axis: 0 Screen Point: 1 Buttons: Hardware Button: 0 Orientation Button: 1 Shake Button: 2	0		3	Control class (byte), control index (byte), handler index (byte)
245(0xf5)	Set control data	Sets parameters for the specified control's operation mode. The control classes and indices are the same as described in command 244. Each mode can have up to 10 data points associated with it. How many should be set and what they should be set to is entirely based on which mode is being used.	0		7	Control class (byte), control index (byte), data point index (byte), data point (float)
246(0xf6)	Get control mode	Reads the handler index of this control's mode. The control classes and indices are the same as described in command 244.	1	Handler index (byte)	2	Control class (byte), control index (byte)
247(0xf7)	Get control data	Reads the value of a certain parameter of the specified control's operation mode. The control classes and indices are the same as described in command 244.	4	Data point (float)	3	Control class (byte), control index (byte), data point index (byte)
248(0xf8)	Set button gyro disable length	Determines how long, in frames, the gyros should be disabled after one of the physical buttons on the sensor is pressed. A setting of 0 means they won't be disabled at all. This setting helps to alleviate gyro disturbances cause by the buttons causing small shockwaves in the sensor.	0		1	Number of frames (byte)
249(0xf9)	Get button gyro disable lentgh	Returns the current button gyro disable length.	1	Number of frames (byte)	0	
250(0xfa)	Get button state	Reads the current state of the sensor's physical buttons. This value returns a byte, where each bit represents the state of the sensor's physical buttons.	1	Button state (byte)	0	
251(0xfb)	Set mouse absolute/relative mode	Puts the mode in absolute or relative mode. This change will not take effect immediately and the sensor must be reset before the mouse will enter this mode. The only parameter can be 0 for absolute (default) or 1 for relative	0		1	Absolute or relative mode (byte)
252(0xfc)	Get mouse absolute/relative mode	Return the current mouse absolute/relative mode. Note that if the sensor has not been reset since it has been put in this mode, the mouse will not reflect this change yet, even though the command will.	1	Absolute or relative mode (byte)	0	
253(0xfd)	Set joystick and mouse present/removed	Sets whether the joystick and mouse are present or removed. The first parameter is for the joystick, and can be 0 for removed or 1 for present. The second parameter is for the mouse. If removed, they will not show up as devices on the target system at all. For these changes to take effect, the sensor driver may need to be reinstalled.	0		2	Joystick present/removed (byte), Mouse present/removed (byte)
254(0xfe)	Get joystick and mouse present/removed	Returns whether the joystick and mouse are present or removed.	2	Joystick present/removed (byte), Mouse present/removed (byte)	0	

Appendix

USB Connector

The 3-Space Sensor has a 5-pin USB Type-B jack and can be connected via a standard 5-pin mini USB cable.

Hex / Decimal Conversion Chart

		Second Hexadecimal digit															
		0	***1***	***2***	***3***	***4***	***5***	***6***	***7***	***8***	***9***	***A***	***B***	***C***	***D***	***E***	***F***
First Hexadecimal Digit	***0***	000	001	002	003	004	005	006	007	008	009	010	011	012	013	014	015
	1	016	017	018	019	020	021	022	023	024	025	026	027	028	029	030	031
	2	032	033	034	035	036	037	038	039	040	041	042	043	044	045	046	047
	3	048	049	050	051	052	053	054	055	056	057	058	059	060	061	062	063
	4	064	065	066	067	068	069	070	071	072	073	074	075	076	077	078	079
	5	080	081	082	083	084	085	086	087	088	089	090	091	092	093	094	095
	6	096	097	098	099	100	101	102	103	104	105	106	107	108	109	110	111
	7	112	113	114	115	116	117	118	119	120	121	122	123	124	125	126	127
	8	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143
	9	144	145	146	147	148	149	150	151	152	153	154	155	156	157	158	159
	A	160	161	162	163	164	165	166	167	168	169	170	171	172	173	174	175
	B	176	177	178	179	180	181	182	183	184	185	186	187	188	189	190	191
	C	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207
	D	208	209	210	211	212	213	214	215	216	217	218	219	220	221	222	223
	E	224	225	226	227	228	229	230	231	232	233	234	235	236	237	238	239
	F	240	241	242	243	244	245	246	247	248	249	250	251	252	253	254	255

Notes:

Serial Number: ______________________________________

Yost Labs

630 Second Street
Portsmouth, Ohio 45662

Phone: 740-876-4936

www.YostLabs.com