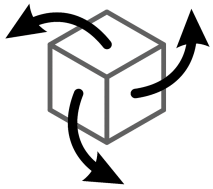
3-Space Sensor



3-Space Sensor Wireless 2.4GHz

Miniature Wireless Attitude & Heading Reference System

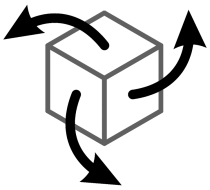
User's Manual

YEI Technology
630 Second Street
Portsmouth, Ohio 45662

www.YeiTechnology.com
www.3SpaceSensor.com

This page intentionally left blank

This page intentionally left blank



3-Space Sensor Wireless 2.4GHz

Miniature Wireless Attitude & Heading Reference System

User's Manual

YEI Technology
630 Second Street
Portsmouth, Ohio 45662

www.YeiTechnology.com
www.3SpaceSensor.com

Toll-Free: 888-395-9029
Phone: 740-355-9029

Table of Contents

This page intentionally left blank

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 Wireless 3-Space Sensor uses RF communication technology, communication failure modes should be carefully considered when designing a system that uses the wireless 3-Space Sensor.
- The Wireless 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

Limited Product Warranty: YEI 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. No warranties exist for any misuse. YEI will repair or replace any defective product which is returned within this time period.

Product Support: YEI provides technical and user support via our toll-free number (888-395-9029) and via email (support@YostEngineering.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 Regulatory Approval

1.3.1 United States FCC Approval

This device contains FCC ID: OA3MRF24J40MA

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses and can radiate radio frequency energy, and if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation. If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna.
- Increase the separation between the equipment and receiver.
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected.
- Consult the dealer or an experienced radio/TV technician for help.

To satisfy FCC RF Exposure requirements for mobile and base station transmission devices, a separation distance of 20 cm or more should be maintained between the antenna of this device and persons during operation. To ensure compliance, operation at closer than this distance is not recommended. The antenna(s) used for this transmitter must not be co-located or operating in conjunction with any other antenna or transmitter.

If the Wireless Unit is used in a portable application (antenna is less than 20 cm from persons during operation), the integrator is responsible for performing Specific Absorption Rate (SAR) testing in accordance with FCC rules 2.1091

1.3.2 Canada IC Approval

This device contains IC ID: 7693A-24J40MA

This device has been certified for use in Canada under Industry Canada (IC) Radio Standards Specification (RSS) RSS-210 and RSS-Gen.

1.3.3 European Approval

The device contains a communication module that has been certified for use in European countries.

The following testing has been completed:

Test standard ETSI EN 300 328 V1.7.1 (2006-10):

- Maximum Transmit Power
- Maximum EIRP Spectral Density
- Frequency Range
- Radiated Emissions

Test standards ETSI EN 301 489-1:2008 and ETSI EN 301 489-17:2008:

- Radiated Emissions
- Electro-Static Discharge
- Radiated RF Susceptibility

1.4 Battery Safety Considerations

The Wireless 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 Wireless 3-space Sensor has been designed to include multiple levels of battery safety assurance. The Wireless 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 YEI technical support for battery replacement or battery repair.
- Don’t let the mobile device or battery come in contact with water.
- 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 YEI 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 YEI Wireless 3-Space Sensor

2.1 Introduction

The YEI 3-Space Sensor™ Wireless integrates a miniature, high-precision, high-reliability, Attitude and Heading Reference System (AHRS) with a 2.4GHz DSSS communication 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 YEI 3-Space Sensor Wireless system also utilizes a dynamic sensor confidence algorithm that ensures optimal accuracy and precision across a wide range of operating conditions.

The YEI 3-Space Sensor Wireless unit features are accessible via a well-documented open communication protocol that allows access to all available sensor data and configuration parameters using either 2.4GHz DSSS wireless or USB 2.0 interfaces. 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 YEI Wireless 3-Space Sensor™ communicates with a host PC via a USB dongle installed in the PC. Up to 15 sensor units can be associated with each wireless dongle, and multiple dongles can be used simultaneously to achieve high sensor counts or increase individual sensor throughput. Sensor and dongle units have individual wireless network PAN Id assignment and wireless channel assignment to allow multiple sensors to communicate simultaneously without interference or performance degradation.

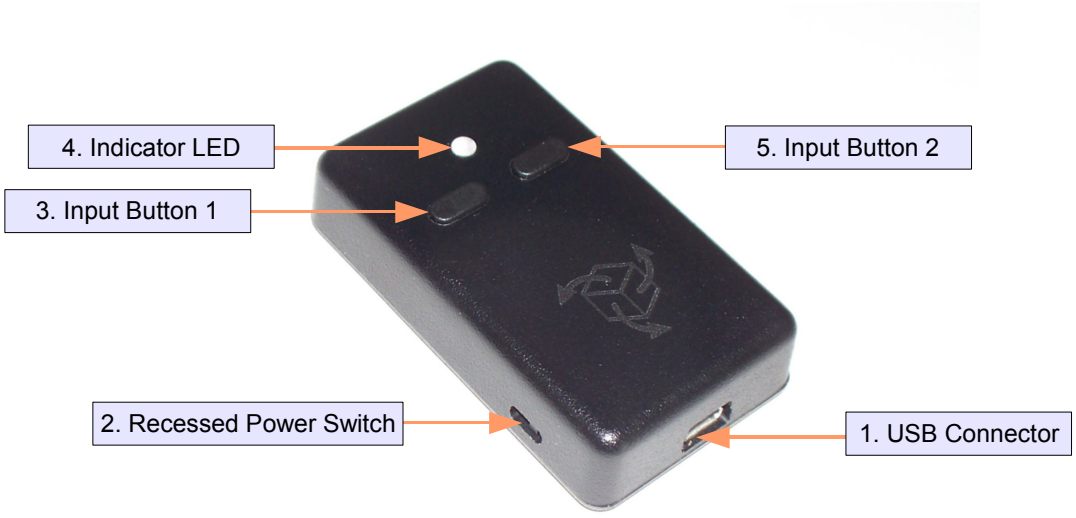
When used as a USB device, the 3-Space Sensor™ provides mouse emulation and joystick emulation modes that ease integration with existing applications.

2.2 Applications

- Robotics
- Motion capture
- Positioning and stabilization
- Vibration analysis
- Inertial augmented localization
- Personnel / pedestrian navigation and tracking
- Unmanned air/land/water vehicle navigation
- Education and performing arts
- Healthcare monitoring
- Gaming and motion control
- Accessibility interfaces
- Virtual reality and immersive simulation

2.3 Hardware Overview

2.3.1 Wireless Sensor 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.

2.3.2 Wireless Dongle Hardware Overview

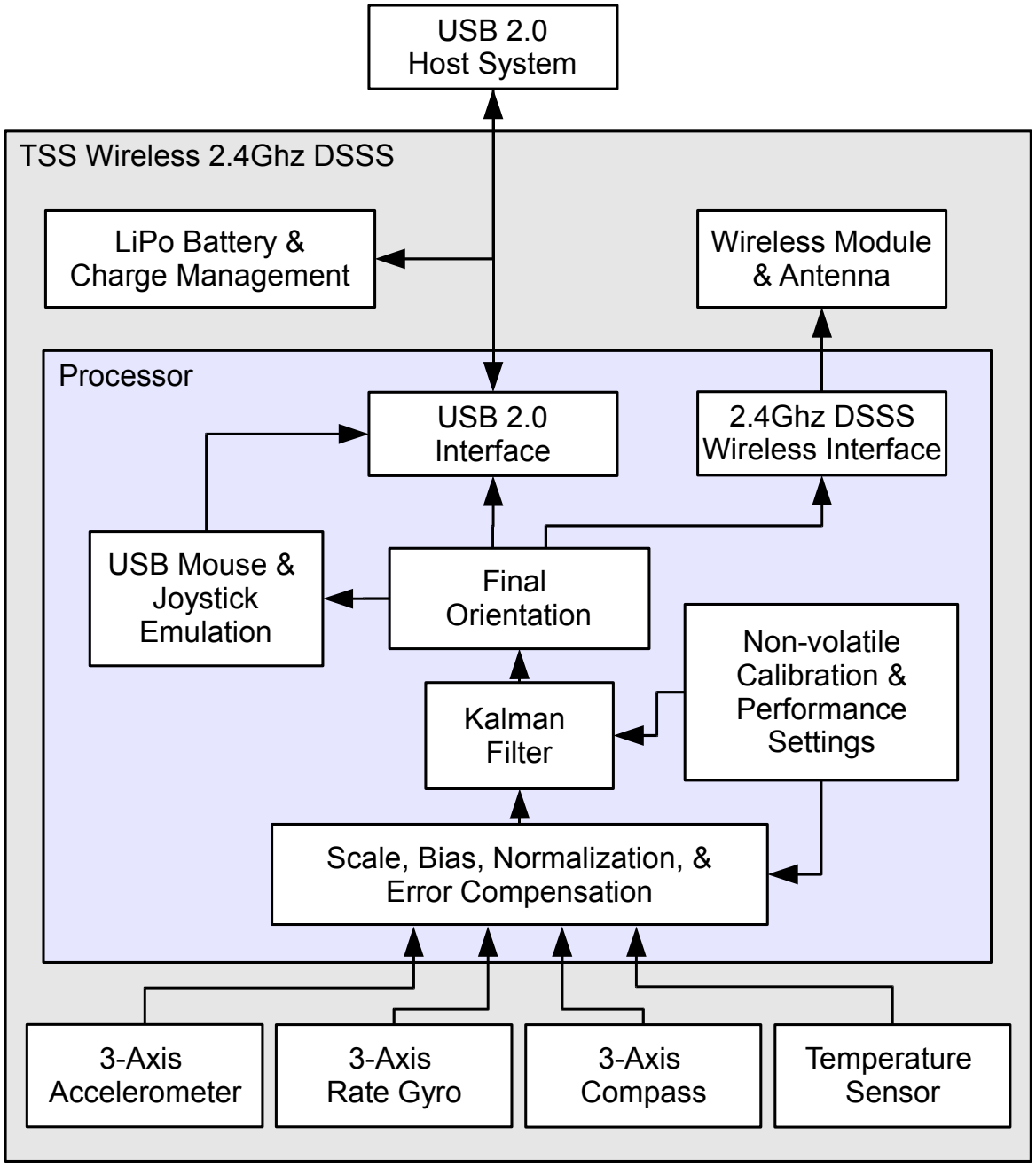
1. **USB Connector** – The 3-Space Wireless Dongle uses a 5-pin mini USB connector to connect to a computer via USB. The USB connector provides for both power and communication.
2. **Indicator LED** – The 3-Space Wireless Dongle includes an RGB LED that can be used for visual status feedback.

2.4 Features

The YEI 3-Space Sensor Wireless 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 wireless AHRS at 35mm x 60mm x 15mm and 28 grams
- Integrated 2.4GHz DSSS wireless communication interface allows high-performance at ranges up to 200'
- Integrated Rechargeable Lithium-Polymer battery and charge control allows battery life of 5+ hours at full performance
- Fast sensor update and filter rate allow use in real-time applications, including stabilization, virtual reality, real-time immersive simulation, and robotics
- 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
- Flexible communication options: USB 2.0 or wireless 2.4GHz DSSS (FCC Certified)
- 2.4Ghz DSSS wireless communication allows orientation sensing without any wires, making activities requiring a high level of mobility like motion capture possible.
- Wireless sensors have configurable wireless channel selection and network PAN Ids to allow multiple sensors to communicate simultaneously without interference or performance degradation
- Each communication dongle unit supports up to 15 independent sensor units
- Asynchronous communication support for improved performance with multiple sensor units
- Communication through a virtual COM port
- 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



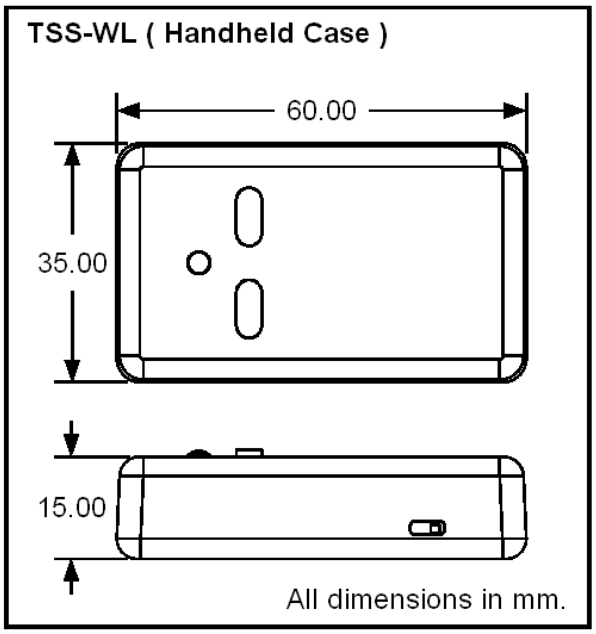
2.6 Specifications

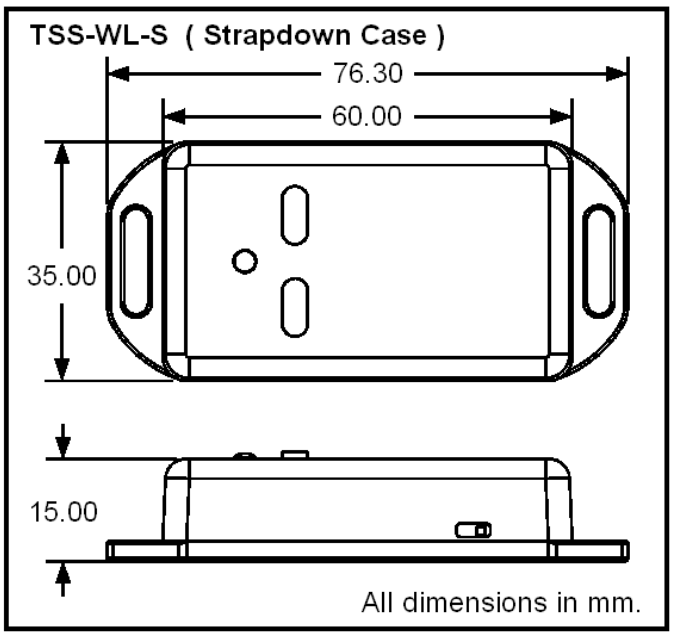
General	
Part number	TSS-WL (Handheld Sensor Unit) TSS-WL-S (Strapdown Sensor Unit)
Dimensions	35mm x 60mm x 15mm (1.38 x 2.36 x 0.59 in.)
Weight	28 grams (0.98 oz)
Supply voltage	+5v USB
Battery technology	rechargeable Lithium-Polymer
Battery lifetime	5+ hours continuous use at full performance
Communication interfaces	USB 2.0, 2.4GHz DSSS Wireless (FCC certified)
Wireless communication range	up to 200'
Wireless PAN Ids selectable	65536
Wireless channels selectable	16 (2.4GHz channel 11 through 26)
Filter update rate	up to 200Hz with full functionality
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
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 0.00048g/digit for ±4g range 0.00096g/digit for ±8g range
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

Dongle	
Part number	TSS-DNG (Wireless Communication Dongle)
Dimensions	22.5mm x 65.6mm x 15mm (0.86 x 2.58 x 0.59 in.)
Weight	12 grams (0.42 oz)
Supply voltage	+5v USB
Communication interfaces	USB 2.0, 2.4GHz DSSS Wireless (FCC certified)
Wireless communication range	up to 200'
Wireless sensors supported	15 simultaneous
Wireless PAN Ids selectable	65536
Wireless channels selectable	16 (2.4GHz channel 11 through 26)
Processor	32-bit RISC running @ 60MHz

*Specifications subject to change

2.7 Physical Dimensions





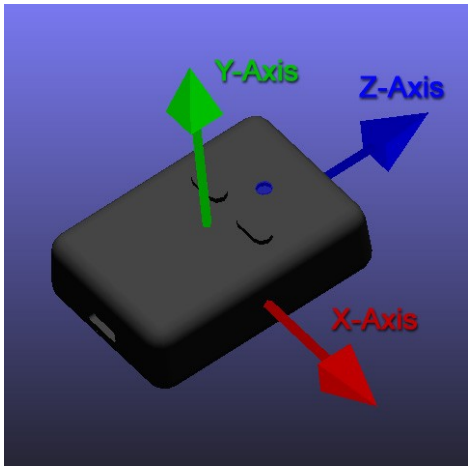
2.8 Axis Assignment

All YEI 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.

2.9 Wireless Terminology

The following provides a list of commonly used wireless concepts and their definitions.

Pan ID – Refers to a 16-bit number that can be assigned to each individual wireless unit or dongle. The pan ID serves the purpose of separating units into clusters or networks, such that a unit with one pan ID cannot communicate with a unit on another pan ID.

Channel – Refers to the frequency on which a given unit transmits or receives upon. There are 16 available channels, ranging from 11-26, inclusive. Certain channels may be more well-suited for wireless communication than others at any given time. Refer to the command listing to find out how to scan channels. Like the pan ID, units with different channels cannot communicate with each other.

Address – Each unit has a unique built-in and unchangeable address (also referred to as hardware ID), which can be found etched into the back of wireless units (but not dongles). When communicating with a unit, addresses are not used directly. Instead, a mapping is provided in the form of logical IDs.

Logical ID – Since direct addresses are cumbersome, these are provided as a means to easily communicate with a given unit. There are 15 such logical IDs. Each logical ID can be mapped to a hardware address to ease communication. A table of logical IDs and their corresponding hardware addresses can be found inside the suite under the Dongle submenu, under Wireless Communication Settings... For more information on reading or setting logical IDs, please refer to the command chart.

2.10 Wireless LED Modes

Both the dongle and wireless unit have built-in LEDs that are meant to convey information about the state of the respective device. Each unit and dongle may also have a custom color that can be set. The wireless unit will display the following LED colors under the following circumstances:

- Upon receipt of a packet, the wireless unit will flash green temporarily. This will occur regardless of whether the wireless unit is plugged in or not.
- When the wireless unit is plugged in and charging, the sensor will flash orange every second.
- When the wireless unit is plugged in and fully charged, the sensor will flash green every second.
- When the wireless unit falls below a certain battery life level, it will flash red in increasingly quicker intervals. Note that this does not happen if the sensor is plugged in.
- Upon receipt of a packet, the dongle will flash green temporarily.
- If the dongle transmits a packet that does not reach its destination, the dongle will flash red temporarily.

Under all other circumstances, both devices will display the custom color that has been set. In addition to this default behavior, it is possible to set a static LED mode, in which the above functionality will be overridden. In this case, the LED will display only the custom color and nothing else. Please refer to the command chart for information on setting static LED mode.

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 subtracted from 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 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.

3.1.4 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.

3.1.5 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.
- **IMU Mode:** Performs no orientation filtering, but allows IMU data to be read at the maximum update rate of 800 Hz.

3.1.6 Reference Orientation/Taring

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.

3.1.7 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.
- **Rho Values:** As mentioned earlier, by default the accelerometer and compass are trusted less than the gyros when the sensor is in motion. Rho values are the mechanism that handles the concept of trust. They 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. A lower value for the parameter means more trust. The default mode for this is “confidence mode”, where the rho value chooses between a minimum and maximum value based on how much the sensor is moving. The other option is to have a single, static rho value.

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 either the USB port or wireless interface, using the 3-Space Wireless Dongle. 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.

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 when plugged in through USB. 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.

3.3.4 Wireless Joystick/Mouse

The 3-Space Dongle can be set up to receive joystick and mouse data from a 3-Space Sensor wirelessly and present this data to the computer via a USB interface. This is accomplished by supplying the logical ID of the wireless device that will act as the mouse/joystick. Commands 240 and 241 are used to enable the wireless joystick and mouse respectively. When either of these commands are invoked, the chosen wireless sensor will immediately begin transmitting the requested HID data to the dongle. The update rate at which this information is received is determined by command 215. Additionally, HID information may be sent synchronously or asynchronously from the wireless sensor to the dongle. Command 217 allows the user to set the desired mode. Synchronous HID mode is the default mode, in which the dongle automatically asks for the requested data first. This mode enjoys a high rate of reliability and it is quite easy to interlace regular protocol commands with HID data transmission/reception. This mode is slower, however, than asynchronous mode, since information must both be requested and received. Asynchronous mode, on the other hand, forces the sensor to automatically send HID information without being asked to do so by the dongle. This allows for much higher update rates, at the expense of reliability due to the increased number of wireless transmissions and potential collisions. It is recommended to use this mode only if you will be using the 3-Space Sensor only as an HID joystick or mouse at the given time.

3.4 Sensor Settings

3.4.1 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.

3.4.2 Committing Wireless Settings

In addition to committing sensor settings, there are also settings specific to wireless devices. In order to commit these settings, command 197 must be used. Note that committing the default settings will have no effect on wireless settings, while committing wireless settings will not change the default settings. A list of wireless settings for the sensor can be found in table 3.4.6 and a list of wireless settings for the dongle can be found in table 3.4.7.

3.4.3 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 of the natural axes.

3.4.4 Sensor General Settings

Setting Name	Purpose	Default Value
Accelerometer Rho Value	Determine how trusted the accelerometer is	Confidence Mode, 5 to 100
Compass Rho Value	Determine how trusted the compass is	Confidence Mode, 5 to 100
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)
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
Reference Mode	Determines how the accelerometer and compass reference vectors are determined	Single Auto
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

3.4.5 Dongle General Settings

Setting Name	Purpose	Default Value
LED Color	Determines the RGB color of the LED	0,0,1(Blue)
Desired Update Rate	Determines how long each cycle should take (ideally)	0 microseconds
LED Mode	Determines whether the LED mode is static or not.	0 (Non-static)

3.4.6 Sensor Wireless Settings

Setting Name	Purpose	Default Value
PanID	Determines the panID of this sensor.	1
Address	Determines the address of this sensor.	Factory determined (cannot be set, only read)
Channel	Determines the channel of this sensor.	26

3.4.7 Dongle Wireless Settings

Setting Name	Purpose	Default Value
PanID	Determines the panID of this dongle.	1
Address	Determines the address of this dongle.	Factory determined (cannot be set, only read)
Channel	Determines the channel of this dongle.	26
Logical ID Table	Determines the mapping between logical ID and addresses.	Array of 15 unsigned 16-bit integers, values initialized to 0
Retries	Determines number of retries dongle will attempt on failed transaction	3
Joystick Logical ID	Determines the logical ID of the device that will act as the joystick, or -1 if there is no joystick desired.	-1
Mouse Logical ID	Determines the logical ID of the device that will act as the mouse, or -1 if there is no mouse desired.	-1
HID Update Rate	Update rate for requesting joystick/mouse information, in milliseconds.	15 (67 hz)
HID Asynchronous Mode	Determines whether joystick/mouse data transmission is asynchronous.	0
Asynchronous Flush Mode	Determines whether or not asynchronously requested data is automatically flushed or whether it must be requested via a dongle command.	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(USB)

When interfacing with a computer through USB, the 3-Space Sensor presents itself as a COM port, which provides a serial interface through which host may communication with the sensor unit by using protocol messages. 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 easiest way to find out which COM port belongs to a certain sensor is to take note of what COM port appears when that sensor is plugged in(provided the drivers have been installed on that computer already. Otherwise, find out what COM port appears once driver installation has finished.) Additionally, each sensor can be identified programatically by reading the serial number of each attached sensor. For more information on how to install the sensor software on a computer and begin using it, see the Quick Start guide.

4.1.3 Computer Interfacing Overview(Wireless)

To interface to a sensor through a computer wirelessly, the 3-Space Dongle must be connected to the computer through USB. The Dongle will present itself as a COM port just as the 3-Space Sensor does. Each dongle can be associated with up to 15 wireless sensor units. To associate a sensor unit with a dongle, the user must place the desired sensor's serial number in one of the dongle's 15 logical wireless table slots. Any wireless 3-Space Sensors in range that have been given an address slot on the Dongle may then be communicated to using the Dongle. For information on how to set up the Dongle's address slots, see the Quick Start guide or <Dongle slot command ##>. For information on what data to send to the Dongle to communicate with a particular sensor, see section 4.3. The wireless communication protocol and wired communication protocol support the same commands, but are not identical. This allows the wireless protocol to include features that are specific to the nature of wireless communication such as wireless addressing, wireless reliability, and packet-loss handling, etc. For more information pertaining to the wired and wireless communication protocols, see sections 4.2 and 4.3 respectively.

4.2 Wired 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

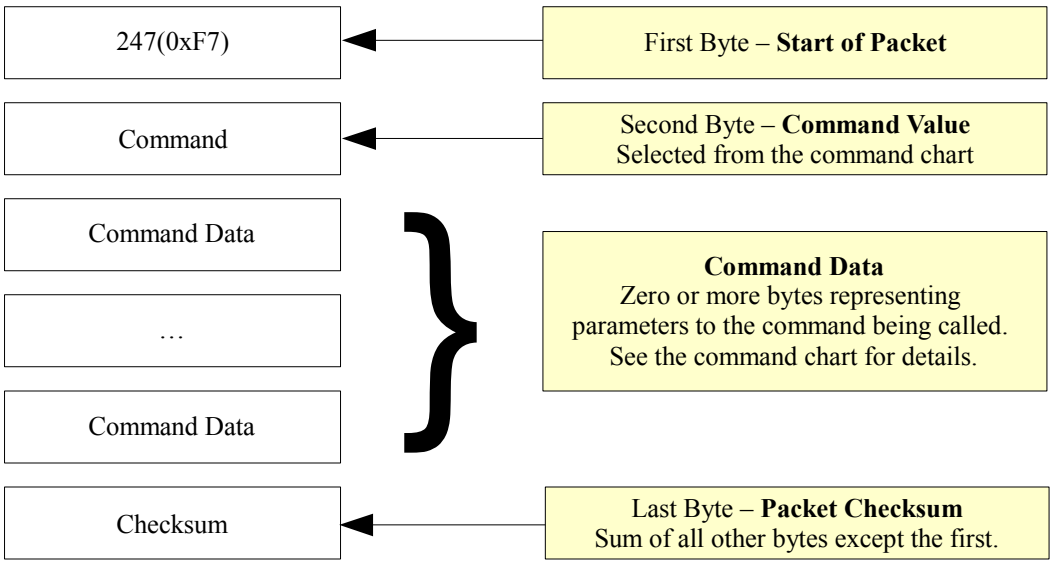


Figure 1 - Binary Command Packet Format

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.

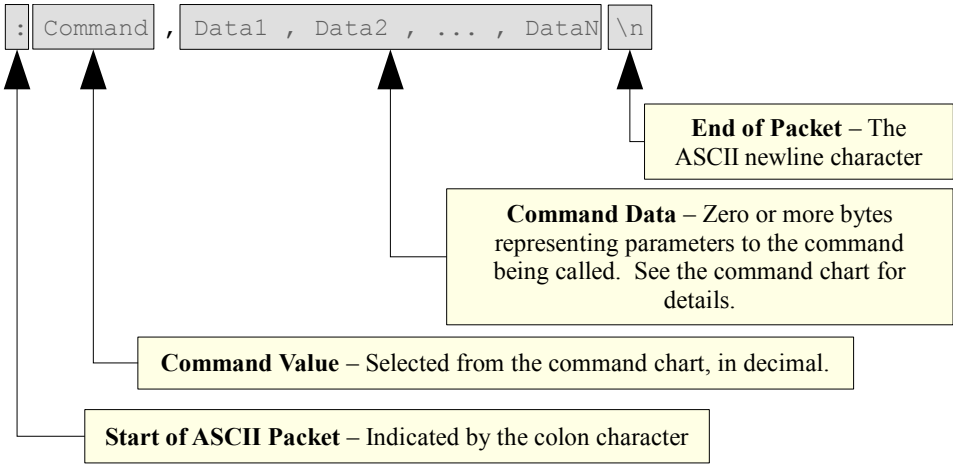


Figure 2 - ASCII Command Packet Format

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. Also note that when communicating with the dongle or sensor in the 3-Space Suite, the newline is automatically appended to the input, thus it is not necessary to add it.

Sample ASCII commands:

`:0\n`	`(If connected to the sensor)`	Read orientation as a quaternion
`:106,2\n`	`(If connected to the sensor)`	Set oversample rate to 2
`:214\n`	`(If connected to the dongle)`	Read signal strength of most recent dongle reception
`:208,5\n`	`(If connected to the dongle)`	Read the hardware ID/address of the unit mapped to logical ID 5

ASCII Response:

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 Wireless Protocol Packet Format

4.3.1 Wireless Communication Format

The protocol for communicating with sensors wirelessly is very similar to the wired protocol, but includes accommodations for wireless unit addressing and wireless communication failures. Thus, all wireless communication messages now also include an address specifying which sensor they are to be sent to. Additionally, each wireless protocol command returns status information pertaining to the success or failure of the wireless command. Any commands sent to address 254 will be sent to the dongle itself.

4.3.2 Binary Packet Format

The wireless binary packet format is very similar to the wired format. Each packet consists of an initial **"address"** 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 wireless binary packet is at least 3 bytes in length and is formatted as shown in figure 3

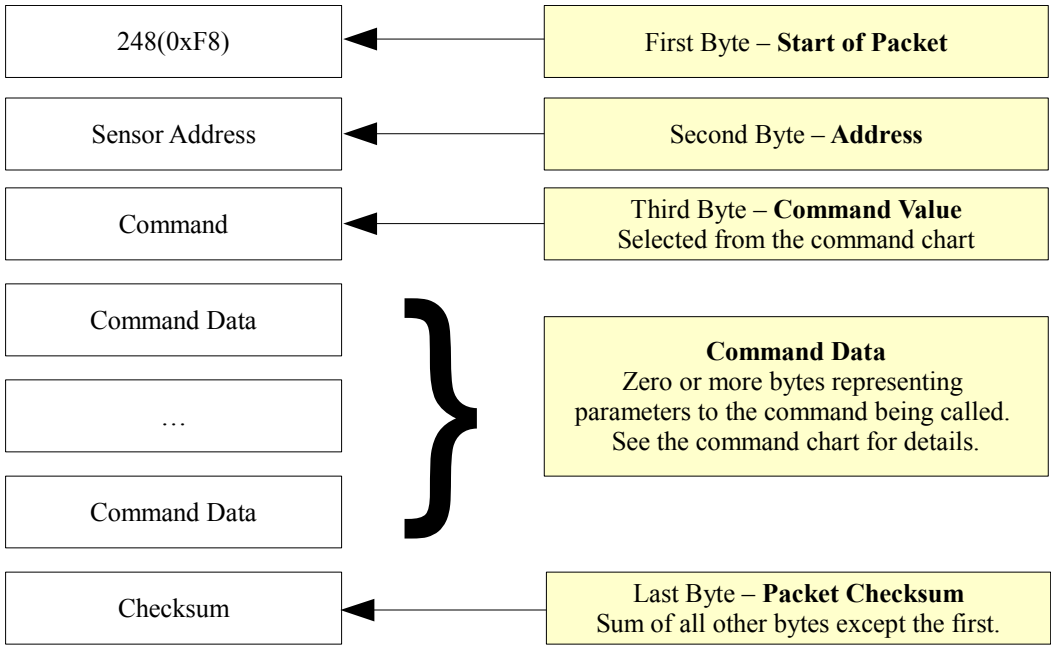
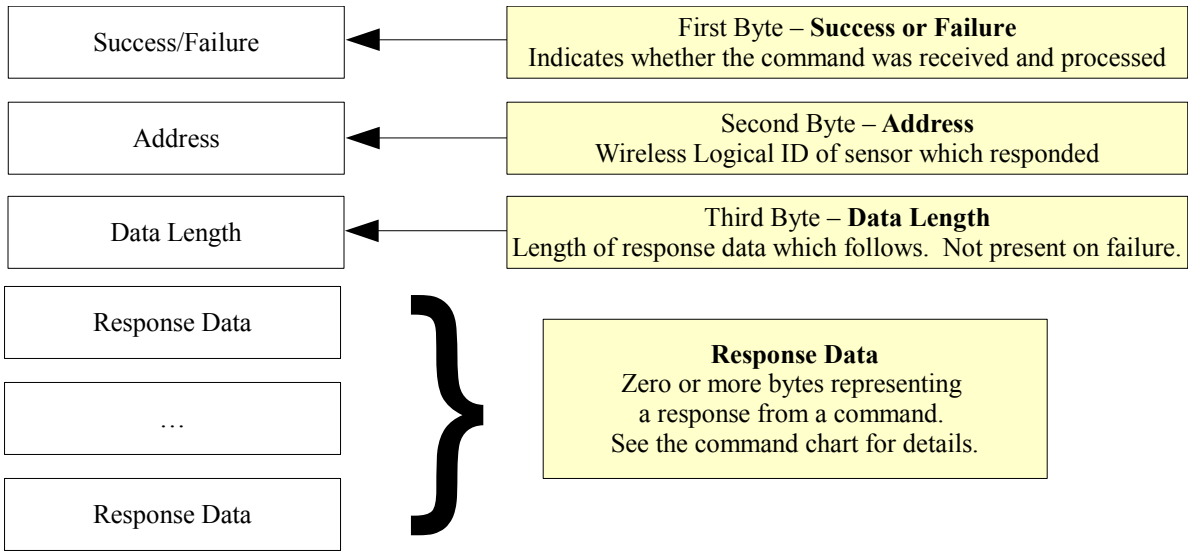


Figure 3 - Wireless Binary Command Packet Format

4.3.3 Binary Command Response



When a binary command is invoked wirelessly, before the data it would normally return in wired mode, it will return status bytes. First is the **success byte**, which is a 0 if the command was successful and non-0 if it was not. Some things which can cause a failure are:

- The lack of corresponding wireless sensor at the specified address.
- Wireless communication errors or dropped packets.
- Improper command formatting or data length

Second is the **address byte**. This indicates which sensor sent the response. If the success byte indicates a success, the third byte, the **data length byte**, will be present as well, indicating how much data follows it. All of the remaining data can be retrieved by reading the number of bytes indicated by **data length**. Keep in mind also that when the dongle is communicated to as if it were a wireless sensor, it will return data in the same format as a wireless response. Since all dongle commands that are formatted properly return success and the dongle address is a constant 254, all dongle responses begin with 00 FE.

4.3.4 Sample Binary Commands

Command	Description	Potential Response
F8 01 00 00	Read orientation as a quaternion from sensor 1	00 01 10 00 00 00 00 00 00 00 00 00 00 00 00 3F 80 00 00
F8 05 6A 02 6C	Set oversample rate to 2 on sensor 5	00 05 00
F8 03 E6 E6	Read version string from sensor 3	00 03 0C 54 53 53 57 49 52 30 36 30 31 31 31
F8 0D EC EC	Read clock speed from sensor 13	00 0D 04 03 93 87 00
F8 09 77 00 00 00 00 BF 80 00 00 00 00 00 00	Set accelerometer reference vector to (0.0, -1.0, 0.0) on sensor 9	00 09 00
F8 FE C0 BE	Read the panID from the dongle	00 FE 02 00 01
F8 FE D7 14 E9	Set HID update rate to 20ms	00 FE 00

4.3.5 ASCII Text Packet Format

Wireless ASCII packets are very similar to wired ASCII packets. Each wireless ASCII packet is formatted as shown in figure 4.

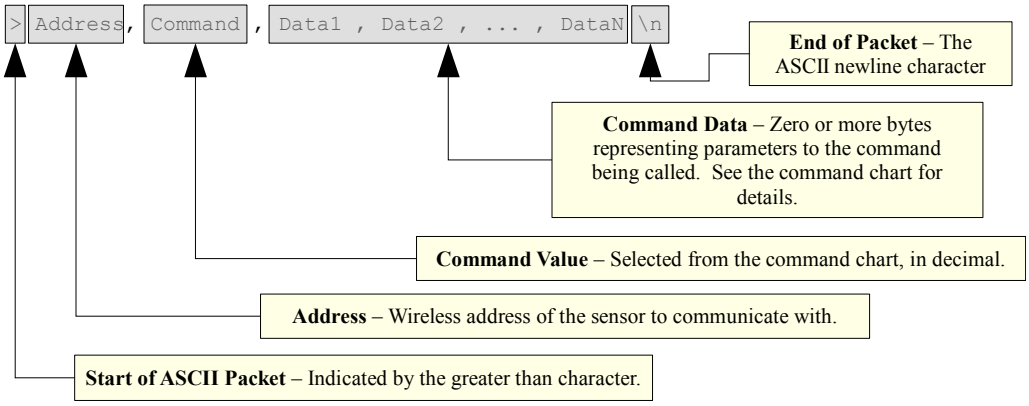


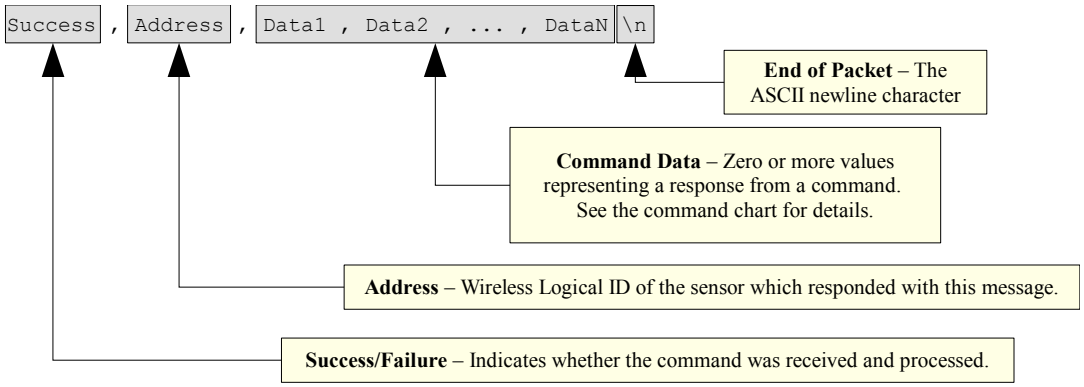
Figure 4 - Wireless ASCII Command Packet Format

Thus the ASCII packet consists of the the following characters:

- **>** – the ASCII greater than 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.

4.3.6 ASCII Command Response



When an ASCII command is called wirelessly, before the data it would normally return in wired mode, it will return status values, each seperated by a comma. First is the **success/failure value**, which is a 0 if the command was successful and 1 if it was not. Some things which can cause a failure are:

- The lack of a sensor present wirelessly
- Communication interference causing the wireless sensor to not respond
- Improper command formatting or data length

Second is the **address**. This indicates which sensor sent the response. So for the command :7,0\n, the response might be 0,7,0.0,0.0,0.0,1.0\n. Failures will only contain the status values and no data, so a failure of the above command would look like 1,7\n.

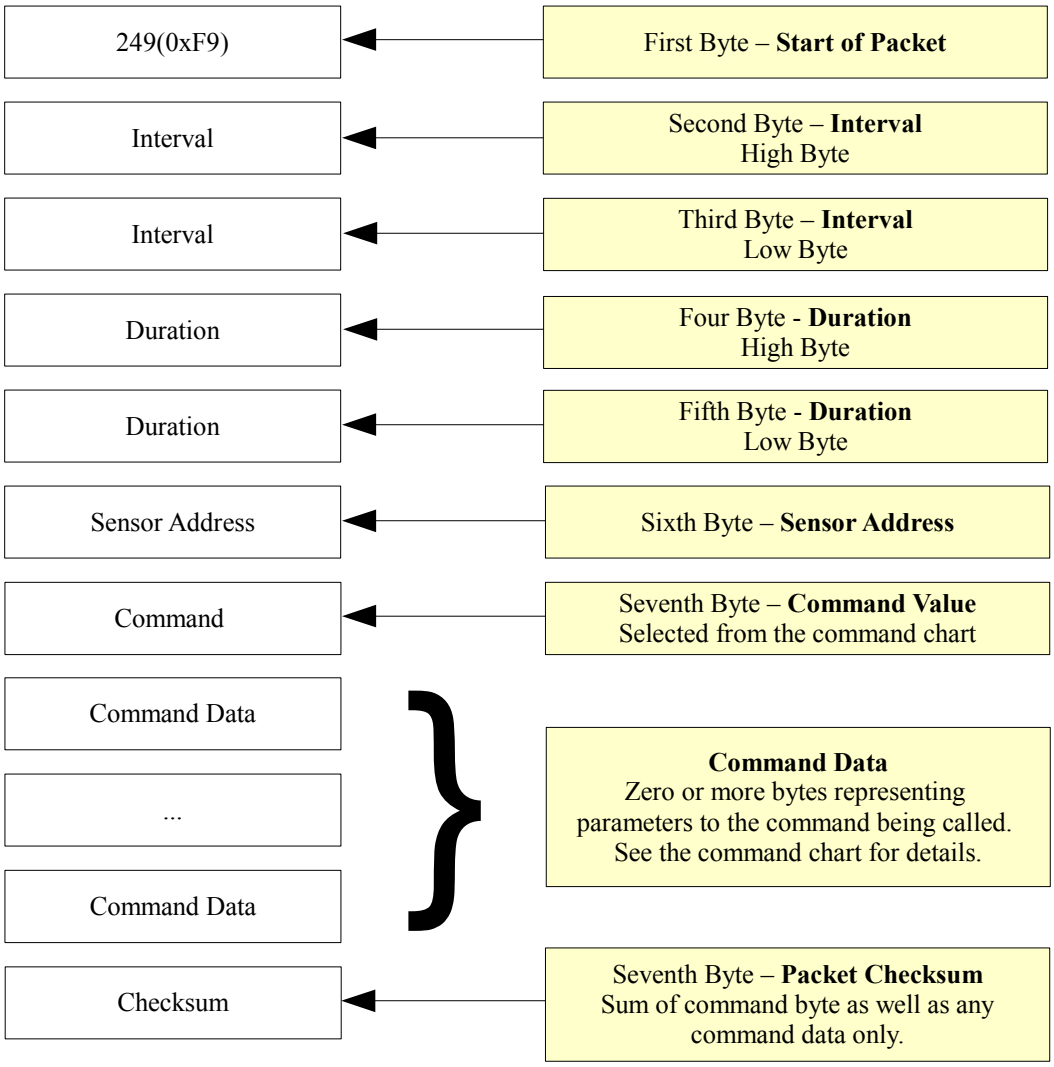
4.3.7 Sample ASCII Commands

Command	Description	Potential Response
>0,1\n	Read orientation as a quaternion from sensor 1	0,1,0.0,0.0,0.707,0.707\n
>5,106,2\n	Set oversample rate to 2 on sensor 5	0,5\n
>3,230\n	Read version string from sensor 3	0,3,TSSWIR060111\n
>13,236\n	Read clock speed from sensor 13	0,13,60000000\n
>9,119,0.0,-1.0,0.0\n	Set accelerometer reference vector to (0.0, -1.0, 0.0) on sensor 9	0,9\n
>254,210\n	Read wireless channel strengths from dongle	0,254,0,4,0,0,0,4,0,1,0,2,0,1,4,0,1,0\n
>254,196,1\n	Set LED mode to static on the dongle	0,254\n

4.4 Wireless Asynchronous Protocol

4.4.1 Asynchronous Communication Format

In addition to the standard request/response communication paradigm, the 3-Space sensor also supports an asynchronous protocol. All asynchronous requests are initiated through the dongle and sent to the corresponding sensor, but only once. This effectively halves communication overhead allowing a wireless sensor to automatically respond with given data at the specified interval for a specified amount of time. Providing that the command format is correct, the sensor given by the specified address will begin automatically transmitting the requested data upon receipt of the dongle's request. Asynchronous requests follow the same format as regular commands, in that the dongle will return status bytes indicating whether or not the asynchronous request was received by the sensor in question. Following is the general format for asynchronous requests:



4.4.2 Interval and Duration

Both the interval and duration are 16-bit unsigned integers representing, in milliseconds, how often data will be sent, and for how long data will be sent. An interval of zero can be specified, which will force the sensor to send the information as fast as possible. Specifying a 0xFFFF for the duration will result in an indefinite duration, while a duration of zero will terminate the asynchronous transmissions altogether for the requested sensor.

4.4.3 Starting/Stopping Asynchronous Transmissions

Since asynchronous requests return status bytes as a normal command would, it is possible to tell whether or not the asynchronous request was acknowledged. Just like any other command, the request itself will return a status code, the logical ID of the sensor, and the number of data bytes. Since these commands return no data, the number of data bytes will always be zero, and in the case of a failure, there will be no data byte present. Since there is presumably little wireless traffic occurring while the sensors are being placed into asynchronous mode, most asynchronous start requests will be received without issue. However, stopping asynchronous transmissions can be more difficult depending on the number of sensors communicating, since there will be more wireless collisions, more channel noise, and worse, the wireless sensor might even be in the middle of sending an asynchronous data transmission at the same time the dongle is requesting that it stop asynchronous communication. The best approach is to attempt each start and stop a number of times until a success is confirmed. If a failure is read, simply re-send the last asynchronous stop attempt. This can be done in a loop, talking to multiple sensors per iteration. This has the added effect that the remaining asynchronous transmissions become easier to stop the less of them there are.

4.4.4 Asynchronous Auto Flush

By default, once a dongle receives data that has been transmitted asynchronously, the data will be flushed as soon as it is received. For this reason, it is optimal to always be reading out data from the dongle once asynchronous mode has been initiated. The data format is nearly the same as a standard binary data response (see section 4.3.3). The only difference in the two is that the success byte will always read 0 in the case of asynchronous data, since there are no continuous requests, and no concept of failure from the dongle's perspective. Please note that asynchronous requests can only be invoked in a binary format, and that all asynchronous responses are binary data as well. For information on changing the asynchronous flush mode, please refer to dongle command 176 in the command chart.

4.4.5 Asynchronous Manual Flush

The dongle can also be configured to flush out asynchronous data only once requested. Additionally, it is possible to enable timestamps for the requested data, as well as prevent all data from certain sensors from being output by the dongle. Please refer to section 4.4.6 or dongle commands 178, 179, 180 and 181 in the command chart for more information. Once the asynchronous manual flush mode has been configured, there are two possible ways to retrieve the data. In the case that only one sensor is configured to transmit data asynchronously, a single read will most likely suffice. The single asynchronous read can be invoked by sending dongle command 182 followed by the logical ID of the sensor that has been configured to transmit asynchronously. The format for this returned data is slightly different than the auto-flushed data, in that there is no success byte. Instead, the dongle will output the logical ID of the sensor and the size of the requested data, followed by the data itself. If timestamps are enabled, this data size will include the size of the timestamp, which is a 32-bit unsigned integer. Note that it is possible that the data size can be zero. This indicates that the dongle has not received any data since the last time it was polled. Following is a snippet of pseudocode showing how to retrieve data with a single read:

```
Enable asynchronous communication for sensor with logical ID N.
Enable asynchronous manual flush mode.
Issue command 182 with single byte parameter N.
Read 1 byte, store as logicalID. (This should be the same as N)
Read 1 byte, store as dataSize.
If dataSize > 0:
      If timestamps are enabled:
            Read 4 bytes, store as timeStamp
            Read dataSize – 4 bytes, store as requestedData
      else:
            Read dataSize bytes, store as requestedData
```

In the case that the dongle has sent multiple asynchronous requests to multiple units, it is much more efficient to perform a bulk read, instead of calling the single read for each logical ID. The bulk read command can be invoked by issuing command 183. The format of this returned data varies slightly from single reads. The first two bytes that are read

will be the size of all of the returned data (the size is a 16-bit unsigned integer). From there, the next byte will be the logical ID of a sensor that has been enabled for flushing out data. The following byte will be the size of the data from that particular sensor, followed by the data itself. If timestamps are enabled, this data size will include the size of the timestamp, which is a 32-bit unsigned integer in units of milliseconds. Note that it is possible that the data size can be zero. Continue to loop through the remaining bytes until all data has been collected. Following is a snippet of pseudocode showing how to retrieve data with a bulk read:

```
Enable asynchronous communication for all desired sensors N.
Enable asynchronous manual flush mode.
Issue command 183.
Read two bytes, store as clusterSize (Total size of all data).
idx = 0
while idx < clusterSize:
      Read 1 byte, store as logicalID.
      idx += 1
      Read 1 byte, store as dataSize.
      idx += 1
      If dataSize > 0:
            If timestamps are enabled:
                  Read 4 bytes, store as timeStamp
                  Read dataSize – 4 bytes, store as requestedData
            else:
                  Read dataSize bytes, store as requestedData
            idx += dataSize
```

4.4.6 Asynchronous Timestamps and Flush Bits

When requesting asynchronously-transmitted data in manual flush mode, it is possible to add timestamps to the output data, by using command 178—passing a parameter of 1 will enable timestamps, while 0 will disable them. Command 179 will return a value indicating whether or not timestamps are enabled. Timestamps are 32-bit unsigned integers representing an absolute time in microseconds. This provides for nearly 71 minutes of continuous timestamping until the value wraps around. Additionally, you can prevent the dongle from outputting data belonging to certain sensors. This is useful if you are bulk reading, and know with certainty, that you will only be communicating with a certain number of sensors, or a particular set of sensors that are already mapped in the address table in a specific manner. This can be set with command 180. For example, calling command 180 with a parameter of 0 and then another 0 will prevent the dongle from flushing any of the data unit 0 possibly sent—In this case, not even the logical ID or data length fields will be present in the bulk read. Calling the same command with a parameter of 1 will re-enable flushing for sensor 0. Command 181 followed by a parameter, will return the flush bit for the sensor specified by the parameter.

4.4.7 Sample Asynchronous Requests

Request	Description	Potential Response
F9 00 0F 00 64 03 00 03	Place sensor unit 3 into asynchronous transmission mode where it will run command 0 every 15 milliseconds, then transmit the resultant quaternion for 100 milliseconds.	00 03 00
F9 00 00 00 00 09 00 09	Disable asynchronous transmission mode for unit 9.	00 09 00
F9 00 00 FF FF 23 00 23	Attempt to send asynchronous request to non-existent unit 23.	01 23
F9 56 78 FF FF 05 20 25	Place sensor unit 5 into asynchronous transmission mode where it will send all raw sensor data every 22136 milliseconds for an indefinite period of time.	00 05 00

4.5 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.

4.3.1 Orientation Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
0(0x00)	Read tared orientation as quaternion	Returns the filtered, tared orientation estimate in quaternion form.	16	Quaternion (float x4)	0	
1(0x01)	Read tared orientation as euler angles	Returns the filtered, tared orientation estimate in euler angle form	12	Euler Angles (float x3)	0	
2(0x02)	Read tared orientation as rotation matrix	Returns the filtered, tared orientation estimate in rotation matrix form	36	Rotation Matrix (float x9)	0	
3(0x03)	Read tared orientation as axis angle	Returns the filtered, tared orientation estimate in axis-angle form	16	Axis (float x3), Angle (float)	0	
4 (0x04	Read 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)	Read difference quaternion	Returns the difference between the measured orientation from last frame and this frame.	16	Quaternion (float x4)	0	
6(0x06)	Read untared orientation as quaternion	Returns the filtered, untared orientation estimate in quaternion form.	16	Quaternion (float x4)	0	
7(0x07)	Read untared orientation as euler angles	Returns the filtered, untared orientation estimate in euler angle form	16	Euler Angles (float x3)	0	
8(0x08)	Read untared orientation as rotation matrix	Returns the filtered, untared orientation estimate in rotation matrix form	36	Rotation Matrix (float x9)	0	
9(0x09)	Read untared orientation as axis angle	Returns the filtered, untared orientation estimate in axis-angle form	16	Axis (float x3), Angle (float)	0	
10(0x0A)	Read 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)	Read 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)	Read untared 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	North Vector (float x3), Gravity Vector (float x3)	0	

4.3.2 Normalized Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
32(0x20)	Read 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 (Vector x3), Gravity Direction (Vector x3), North Direction (Vector x3)	0	
33(0x21)	Read normalized gyro rate	Returns the normalized gyro rate vector, which is in units of radians/sec.	12	Gyro Rate (Vector x3)	0	
34(0x22)	Read 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)	Read 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	

4.3.3 Other Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
36(0x24)	Read temperature C	Returns the temperature of the sensor in Celsius.	4	Temperature (float)	0	
37(0x25)	Read temperature F	Returns the temperature of the sensor in Fahrenheit	4	Temperature (float)	0	
38(0x26)	Read 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	

4.3.4 Corrected Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
39(0x27)	Read corrected accelerometer	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)	Read corrected compass	Returns the compass vector in units of gauss.	12	Compass Vector in units of gauss (float x3)	0	

4.3.5 Raw Data Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
64(0x40)	Read 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)	Read 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)	Read 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)	Read 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	

4.3.6 Configuration Write Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
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 rho mode	Determines how trusted the accelerometer contribution is to the overall orientation estimation. Higher values mean that the accelerometer is less trusted.	0		4	Accelerometer rho value (float)
100(0x64)	Set confidence accelerometer rho mode	Determines how trusted the accelerometer contribution is to the overall orientation estimation. Instead of using a single value, uses a minimum and maximum value. Rho values will be changed within 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 rho value (float), Maximum accelerometer rho value (float)
101(0x65)	Set static compass rho mode	Determines how trusted the accelerometer contribution is to the overall orientation estimation. Higher values mean that the compass is less trusted.	0		4	Compass rho value (float)
102(0x66)	Set confidence compass rho mode	Determines how trusted the compass contribution is to the overall orientation estimation. Instead of using a single value, uses a minimum and maximum value. Rho values will be changed within this range depending on the confidence factor. This can have the effect of reducing the compass's effect on the overall orientation estimation and thus reducing magnetically-induced interference.	0		8	Minimum compass rho value (float), Maximum compass rho 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), Pin (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 performance. 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 10. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Samples Per Iteration (Byte)
107(0x6B)	Enable/disable gyroscope	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	Mode (Byte)
108(0x6C)	Enable/disable accelerometer	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	Mode (Byte)

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
109(0x6D)	Enable/disable compass	Enable or disable compass readings as inputs to the orientation estimation. Note that compass readings are still accessible via commands. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		1	Mode (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 which axis each of the natural axes will be read as: 000: XYZ (standard operation) 001: XZY 002: YXZ 003: YZX 004: ZXY 005: ZYX (For example, using XZY means that whatever value appears as Y on the natural axes will now be the Z component of any new data and vice-versa.) 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)	0		1	Axis Direction Byte (byte)

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
117(0x75)	Set running average percent	Sets what percentage of running average to use on the sensor's orientation. This is computed as follows: total_orient = total_orient * percent total_orient = total_orient + current_orient * (1 – percent) current_orient = total_orient If the percentage is 0, the running average will be shut off completely. Maximum value is 97%. This setting can be saved to non-volatile flash memory using the Commit Settings command.	0		4	Running Average 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(0x7c)	Reset Kalman filter	Resets Kalman filter's state and covariance matrices.	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, and a 3 places the sensor into Complementary 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)

4.3.7 Configuration Read Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
128(0x80)	Read tare orientation as quaternion	Returns the current tare orientation as a quaternion.	16	Quaternion (float x4)	0	
129(0x81)	Read tare orientation as rotation matrix	Returns the current tare orientation as a rotation matrix.	36	Rotation Matrix (float x9)	0	
130(0x82)	Read accelerometer rho value	Returns the current accelerometer rho mode as well as the value. If this mode is set to 0 (static), this will return the rho mode, the static rho value, and then a dummy value of 0. If this mode is set to 1, this will return the rho mode, and the minimum and maximum rho values.	9	Accelerometer rho mode (byte), Accelerometer rho values (float x2)	0	
131(0x83)	Read compass rho value	Returns the current compass rho mode as well as the value. If this mode is set to 0 (static), this will return the rho mode, the static rho value, and then a dummy value of 0. If this mode is set to 1, this will return the rho mode, and the minimum and maximum rho values.	9	Compass rho mode (byte), Compass rho values (float x2)	0	
132(0x84)	Read 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)	Read 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)	Read 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)	Read 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)		
136(0x88)	Read 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)	Read 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)	Read 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)	Read 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)	Read 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)	Read 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)	Read 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)	Read axis-direction byte	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)	Read oversample rate	Returns a value 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.	1	Oversample rate (byte)	0	
145(0x91)	Read running average percent	Returns a value indicating how heavily the orientation estimate is based upon the estimate from the previous frame. For more information on the meaning of this value, please refer to the Set Running Average Percent command (117).	4	Running average percent (float)	0	
146(0x92)	Read 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	
147(0x93)	Read Kalman filter covariance matrix	Return the current Kalman filter covariance matrix.	36	Covariance matrix (float x9)	0	
148(0x94)	Read 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)	Read multi-reference mode power weight	Read weighting power for multi-reference vector weights. Intended for advanced users.	4	Weight (float)	0	

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
150(0x96)	Read 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)	Read 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)	Read filter mode	Returns the current filter mode, which can be 0 for IMU mode, 1 for Kalman, 2 for Alternating Kalman or 3 for Complementary. For more information, please refer to the Set Filter Mode command (123).	1	Filter mode (byte)	0	
153(0x99)	Read 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)	Read 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)	Read 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	

4.3.8 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	Bias (float x3), Matrix (float x9)
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	Bias (float x3), Matrix (float x9)
162(0xa2)	Read compass calibration coefficients	Return the current compass calibration parameters.	48	Bias (float x3), Matrix (float x9)		
163(0xa3)	Read accelerometer calibration coefficients	Return the current accelerometer calibration parameters.	48	Bias (float x3), Matrix (float x9)		
164(0xa4)	Read gyroscope calibration coefficients	Return the current gyroscope calibration parameters.	48	Bias (float x3), Matrix (float x9)		
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	Bias (float x3), Matrix (float x9)
169(0xa9)	Set calibration mode	Sets the current calibration mode, which can be 0 for 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)	Read 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)	Read 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	

4.3.9 Dongle Wireless Asynchronous Flush Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
176(0xb0)	Set async auto-flush mode	Set the wireless communication's asynchronous flush mode. If this value is set to 0 (default), data must be 'released' using manual flush commands. If this value is set to 1, data will be output immediately via the dongle's USB connection. For more information, refer to Section 4.4 Wireless Asynchronous Protocol. This setting can be set to non-volatile flash memory by using the Commit Settings command.	0		1	Auto-flush mode (byte)
177(0xb1)	Returns the current auto-flush mode	Returns the wireless communication's current asynchronous flush mode, which can be 0 for auto flush and 1 for manual flush. For more information, refer to Section 4.4 Wireless Asynchronous Protocol.	1	Auto-flush mode (byte)	0	
178(0xb2)	Set async timestamping	Allows the dongle to control timestamping information that can be prepended to manually flushed data. Possible values are 0 for no timestamp, or 1 for timestamping. For more information, refer to Section 4.4.6 Asynchronous Timestamps and Flush Bits.	0		1	Timestamping enabled (byte)
179(0xb3)	Get async timestamping	Returns a value indicating whether timestamping is currently enabled on manually flushed data. Possible values are 0 for no timestamp or 1 for timestamping. For more information, refer to Section 4.4.6 Asynchronous Timestamps and Flush Bits.	1	Timestamping enabled (byte)	0	
180(0xb4)	Set async flush bitfield	Allows the dongle to control which wirelessly received data is output via manual flush mode. The parameter represents a bitfield that represents which wireless sensors' logical IDs can currently output data. For more information, refer to Section 4.4.6 Asynchronous Timestamps and Flush Bits.	0		2	Manual flush bitfield (short)
181(0xb5)	Get async flush bitfield	Returns the current manual flush bitfield. For more information, refer to Section 4.4.6 Asynchronous Timestamps and Flush Bits.	2	Manual flush bitfield (short)	0	
182(0xb6)	Manual flush single	Flush data output for a single logical ID. For more information, refer to Section 4.4.5 Asynchronous Manual Flush.	Varies		1	Logical ID (Byte)
183(0xb7)	Manual flush bulk	Flush data output for all logical IDs. For more information, refer to Section 4.4.5 Asynchronous Manual Flush.	Varies		0	

4.3.10 Wireless Sensor & Dongle Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
192(0xc0)	Read wireless panID	Return the current panID for this wireless sensor or dongle. For more information, refer to Section 2.9 Wireless Terminology.	2	PanID (short)	0	
193(0xc1)	Set wireless panID	Set the current panID for this wireless sensor or dongle. Note that the panID for a wireless sensor can only be set via the USB connection. For more information, refer to Section 2.9 Wireless Terminology. This setting can be committed to non-volatile flash memory by calling the Commit Wireless Settings command.	0		2	PanID (short)
194(0xc2)	Read wireless channel	Read the current channel for this wireless sensor or dongle. For more information, refer to Section 2.9 Wireless Terminology.	1	Channel (Byte)		
195(0xc3)	Set wireless channel	Set the current channel for this wireless sensor or dongle. For more information, refer to Section 2.9 Wireless Terminology. This setting can be committed to non-volatile flash memory by calling the Commit Wireless Settings command.	0		1	Channel (byte)
197(0xc5)	Commit wireless settings	Commits all current wireless 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	
198(0xc6)	Read wireless address	Read the wireless hardware address for this sensor or dongle.	2	Address (short)		

4.3.11 Battery Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
201(0xc9)	Read 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)	Read 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.	2	Battery level as percent (short)	0	
203(0xcb)	Read 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	

4.3.12 Dongle Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
208(0xd0)	Read serial number at logical ID	Return the mapped serial number for the given logical ID.	4	Serial number (int)	1	Logical ID (Byte)
209(0xd1)	Set serial number at logical ID	Set the mapped serial number given by the logical ID. This setting can be committed to non-volatile flash memory by calling the Commit Wireless Settings command.	0		5	Logical ID (Byte), Serial number (int)
210(0xd2)	Read wireless channel noise levels	Return the noise levels for each of the 16 wireless channels. A higher value corresponds to a noisier channel, which can significantly impact wireless reception and throughput.	16	Channel strengths (byte x16)	0	
211(0xd3)	Set wireless retries	Set the number of times a dongle will attempt to re-transmit a data request after timing out. Default value is 3. This setting can be committed to non-volatile flash memory by calling the Commit Wireless Settings command.	0		1	Retries (byte)
212(0xd4)	Read wireless retries	Read the number of times a dongle will attempt to re-transmit a data request after timing out. Default value is 3.	1	Retries (byte)	0	
213(0xd5)	Read wireless slots open	The dongle can simultaneously service up to sixteen individual data requests to wireless sensors. As sensors respond, requests are removed from the table. In the case that too many requests are sent to the dongle in too short a period, the dongle will begin tossing them out. This value will return the number of slots currently open. If this value is 0, no more wireless requests will be handled until some are internally processed.	1	Slots open (byte)	0	
214(0xd6)	Read signal strength	Returns a value indicating the reception strength of the most recently received packet. Higher values indicate a stronger link.	1	Last packet signal strength (byte)	0	

4.3.13 General Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
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.	1	LED mode (byte)	0	
200(0xc8)	Read LED Mode	Returns the current sensor LED mode, which can be 0 for standard or 1 for static.	0		1	LED mode (byte)
223(0xdf)	Read 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)	Enable watchdog timer	Enables the onboard watchdog timer with the specified timeout rate. If a frame takes more than this amount of time, the sensor will automatically reset.	0		4	Timeout rate in microseconds (int)
228(0xe4)	Disable watchdog timer	Disables the watchdog timer.	0		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)	Read 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)	Read 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	
235(0xeb)	Set clock speed	Sets the current processor clock speed. Possible values are 15Mhz, 30 Mhz or 60 Mhz (default). This setting does not need to be committed, but does not take effect until the sensor is reset.	0		4	Clock speed in Hz (int)
236(0xec)	Get clock speed	Returns the current processor clock speed.	4	Clock speed in Hz (int)	0	
237(0xed)	Get serial number	Returns the serial number, which will match the value etched onto the physical sensor.	4	Serial number (int)		
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	

4.3.14 Wireless HID Commands

Command	Description	Long Description	Return Data Len	Return Data Details	Data Len	Data Details
215(0xd7)	Set wireless HID update rate	Specify the interval at which HID information is requested by the dongle. The default and minimum value is 15ms in synchronous HID mode. In asynchronous HID mode, the minimum is 5ms. This setting can be committed to non-volatile flash memory by calling the Commit Wireless Settings command.	0	Last packet signal strength (byte)	1	HID update rate in milliseconds (byte)
216(0xd8)	Read wireless HID update rate	Return the interval at which HID information is requested by the dongle.	1	HID update rate in milliseconds	0	
217(0xd9)	Set wireless HID asynchronous mode	Sets the current wireless HID communication mode. Supplying a 0 makes wireless HID communication synchronous, while a 1 makes wireless HID asynchronous. For more information, refer to Section 3.3.4 Wireless Joystick/Mouse. This setting can be committed to non-volatile flash memory by calling the Commit Wireless Settings command.	0		1	HID communication mode (byte)
218(0xda)	Read wireless HID asynchronous mode	Returns the current wireless HID communication mode, which can be a 0 for synchronous wireless HID or a 1 for asynchronous wireless HID.	1	HID communication mode	0	
240(0xf0)	Set joystick logical ID	Causes the sensor at the specified logical ID to return joystick HID data. Passing a -1 will disable wireless joystick data. For more information, refer to Section 3.3.4 Wireless Joystick/Mouse.	0		1	Joystick logical ID (signed byte)
241(0xf1)	Set mouse logical ID	Causes the sensor at the specified logical ID to return mouse HID data. Passing a -1 will disable wireless mouse data. For more information, refer to Section 3.3.4 Wireless Joystick/Mouse.	0		1	Mouse logical ID (signed byte)
242(0xf2)	Read joystick logical ID	Returns the current logical ID of the joystick-enabled sensor or -1 if none exists.	1	Joystick-enabled logical ID (byte)	0	
243(0xf3)	Read mouse logical ID	Returns the current logical ID of the mouse-enabled sensor or -1 if none exists.	1	Mouse-enabled logical ID (byte)	0	

4.3.15 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)	Read joystick enabled	Read whether the sensor is currently streaming joystick HID data.	1	Joystick enabled state (byte)	0	
243(0xf3)	Read mouse enabled	Read whether the sensor is currently streaming mouse HID data.	1	Mouse enabled state (byte)	0	

4.3.16 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)	Read 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)	Read 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)	Read 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)	Read 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: _____

YEI Technology

630 Second Street
Portsmouth, Ohio 45662

Toll-Free: 888-395-9029
Phone: 740-355-9029

www.YeiTechnology.com
www.3SpaceSensor.com