



Goodrich ISR Systems

Crista IMU Communications Specification v1.3.0

March 2, 2011

Authors:

Jon Becker / Jeff Fisher

2621 Wasco Street / PO Box 1500 / Hood River, OR 97031

(541) 387-2120 phone / (541) 387-2030 fax

www.cloudcaptech.com / sales.cct@goodrich.com / support.cct@goodrich.com



Cloud Cap Technology, a Goodrich Company

Table of Contents

1	Introduction.....	6
2	Serial (RS-232) Interface	6
2.1	Standard Serial Packet Format.....	6
1.1.1	Computing the CRC-16 Checksum	7
2.2	High Speed Serial Packet Format	7
3	CAN Interface.....	8
3.1	CAN Message ID (CANnelli 2.0).....	8
4	CAN Packet Format.....	9
4.1	Packet Types.....	9
4.2	RAWGYRO_IMU_MSG - Raw Gyro Readings.....	10
4.3	RAWACCEL_IMU_MSG - Raw Accelerometer Readings.....	10
4.4	TIMING_IMU_MSG - PPS Timing Data	11
4.5	RESOLUTION_IMU_MSG - Engineering Units Output Resolution	11
4.6	RESUNITS_GYRO_IMU_MSG - Converted Gyro Data.....	12
4.7	RESUNITS_ACCEL_IMU_MSG - Converted Accelerometer Data.....	13
4.8	SET_SETTINGS_IMU_MSG - Configure IMU Output Settings.....	13
4.9	SETTINGS_IMU_MSG - IMU Output Settings Data.....	14
4.10	MFRCALDATE_IMU_MSG - Manufacture and Calibration Dates	15
4.11	SERIALNUMCONFIG_IMU_MSG - Hardware Configuration Data.....	15
4.12	SWVERSION_IMU_MSG - Software Version Information	16
4.13	REQ_CONFIG_IMU_MSG - Request IMU Settings	17
4.14	REQ_CALPARAM_IMU_MSG - Request Calibration Parameter	17
4.15	CALPARAM_IMU_MSG - Calibration Parameter Data.....	17
4.16	RAWGYROTEMPX_IMU_MSG - X Gyro Temperature	18
4.17	RAWGYROTEMPY_IMU_MSG - Y Gyro Temperature	18
4.18	RAWGYROTEMPZ_IMU_MSG - Z Gyro Temperature.....	18
4.19	CRC_STATUS_IMU_MSG - Sensor Dead Data Integrity Status	19
4.20	GYRO_STDEV_IMU_MSG - Gyro Oversample Noise.....	19
4.21	ACCEL_STDEV_IMU_MSG - Accelerometer Oversample Noise.....	20
4.22	GYRO_MINIMUM_IMU_MSG - Gyro Minimum Value.....	20

4.23	ACCEL_MINIMUM_IMU_MSG - Accelerometer Minimum Value	20
4.24	GYRO_MAXIMUM_IMU_MSG - Gyro Maximum Value	21
4.25	ACCEL_MAXIMUM_IMU_MSG - Accelerometer Maximum Value	21
4.26	HS_RAWGYROTEMP_IMU_MSG - High Speed Gyro Temperature.....	22
4.27	HS_RAW_IMU_MSG - High Speed Raw IMU Data.....	22
4.28	HS_SERIAL_IMU_MSG - High Speed Converted IMU Data.....	23
5	Raw Data.....	24
5.1	Converting to Engineering Units	24
5.2	Applying Calibration Data.....	24
1.1.2	Interpolating Temperature-Induced Accelerometer Offsets	25
5.3	Cross-Axis and Acceleration Bias Corrections.....	25
6	Serial EEPROM.....	27
6.1	Sensor Calibration Parameters.....	28

Revision History

Version	Date	Comments
1.1.5	February 27, 2004	Initial public release
1.1.7	June 20, 2005	Support for non-default sensor ranges
1.2.0	December 8, 2005	Major performance optimization, support for HS_RAW mode
	July 13, 2006	Documentation for post processing cross-axis corrections
	January 16, 2007	Added cross-axis corrections to EEPROM table
	January 9, 2008	Added new gyro and accelerometer configurations
1.3.0	November 17, 2008	Major restructuring, added on-board cross-axis corrections and sample stats

List of Tables

Table 1 - General Serial Packet Definition	6
Table 2 - CANnelli 2.0 Message Identifier Scheme	8
Table 3 - Packet Type Identifiers.....	9
Table 4 - Raw Gyro Data Message Format	10
Table 5 - Raw Accelerometer Data Message Format	10
Table 6 - GPS Pulse-Per-Second Timing Data Message Format	11
Table 7 - Converted Output Data Resolution Message Format.....	12
Table 8 - Converted Gyro Data Message Format	12
Table 9 - Converted Accelerometer Data Message Format.....	13
Table 10 - IMU Output Settings Configuration Message Format	13
Table 11 - IMU Output Settings Configuration Message Format	14
Table 12 - Manufacture and Calibration Dates Message Format	15
Table 13 - Hardware Configuration Message Format	15
Table 14 - Gyro Configuration Identifiers	16
Table 15 - Accelerometer Configuration Identifiers.....	16
Table 16 - Software Version Information Message Format	16
Table 17 - IMU Settings Request Message Format	17
Table 18 - Calibration Parameter Request Message Format	17
Table 19 - Calibration Parameter Message Format	17
Table 20 - X Gyro Temperature Message Format	18
Table 21 - Y Gyro Temperature Message Format	18
Table 22 - Z Gyro Temperature Message Format	19
Table 23 - Sensor Head CRC Status Message Format	19
Table 24 - Gyro Oversample Noise Message Format.....	19
Table 25 - Accelerometer Oversample Noise Message Format	20
Table 26 - Gyro Minimum Oversample Message Format	20
Table 27 - Accelerometer Minimum Oversample Message Format.....	21

Table 28 - Gyro Maximum Oversample Message Format	21
Table 29 - Accelerometer Maximum Oversample Message Format	21
Table 30 - High Speed Serial Raw Temperature Data Message Format	22
Table 31 - High Speed Serial Raw Sensor Data Message Format.....	23
Table 32 - High Speed Converted Data Serial Packet Format.....	23
Table 33 - Gyro Temperature Channels to Accelerometers	25
Table 34 - EEPROM Memory Layout.....	27
Table 35 - Sensor Calibration Parameter Data Structure Format	28
Table 36 - Sensor Identifier Numbers.....	29

List of Equations

Equation 1 - PPS clock error calculation.	11
Equation 2 - Calculating the engineering unit data conversion factor.....	12
Equation 3 - Online algorithm for estimating sample standard deviations.....	20
Equation 4 - Computing raw temperature conversion factors.	22
Equation 5 - LSB size for the Crista IMU.	24
Equation 6 - Converting A/D counts (<i>C</i>) to volts (<i>V</i>).	24
Equation 7 - Converting volts to engineering units	25
Equation 8 - Sensor orthogonality corrections.....	26
Equation 9 - Removing acceleration bias effects.....	26
Equation 10 - Finding the EEPROM address of a calibration parameter.	29
Equation 11 - Computing the calibration parameter number to request from the IMU.	29

1 Introduction

This document describes both the CAN and RS-232 data protocols used by the Crista IMU, as well as the data available in the sensor head’s on-board EEPROM and the format in which it is stored. There are two available hardware configurations, each with its own methods of communication and data processing:

- **Crista Sensor Head** - Consists of the gyroscopes and accelerometers along with a multiplexing analog-to-digital converter and serial EEPROM. The raw sensor readings and EEPROM data are directly accessible using the SPI protocol.
- **Crista IMU** - An assembly consisting of the Crista Sensor Head and a processor board that processes sensor input from the sensor head appropriately and sends the data at a user-defined rate over using one or both of the CANopen 2.0b and RS-232 protocols.

The Crista Sensor Head uses a Texas Instruments ADS8344 for analog-to-digital conversions and a Microchip 25LC640 as a serial EEPROM. The communications protocols defined in this document will only be those pertaining to the full Crista IMU assembly. To communicate directly with the Crista Sensor Head, refer to the data sheets for more information.

Note: For the purposes of this document, all bits or bytes numbered ‘0’ indicate the greatest significance, meaning that byte or bit n corresponds to the n-th byte or bit received. Also, all multi-byte data types are transmitted in most the significant byte (MSB) first, or big-endian, format.

2 Serial (RS-232) Interface

The Crista IMU serial interface is a standard 3-wire RS-232 full duplex serial port running at 115,200 bits per second with one start and one stop bit, and no parity. The serial interface is used for sensor telemetry as well as for reading and writing internal configuration data.

2.1 Standard Serial Packet Format

Most messages sent to and received from the Crista IMU share a simple format based loosely on the CANopen 2.0b protocol (See *Section 3 - CAN Interface*). Each message contains a length byte, a message identifier and one to eight bytes of data. The messages sent over either datalink (serial or CAN) share an identical data payload format and message number.

Table 1 - General Serial Packet Definition

Byte	Name	Meaning
0	SYNC_0	Synchronization character used to signal that a packet <i>may</i> be forthcoming. Must be 0x55.
1	SYNC_1	Synchronization character used to signal that a packet <i>may</i> be forthcoming. Must be 0xAA.
2	PktType	The unique identifier for this particular packet type. See Table 3 for a complete list of packet type IDs.
3	Size	The number of data bytes to follow.
4...Size+3	Data	Data payload.
Size+5	CRC16_0	Most and least significant bytes of the CRC-16 checksum. The IMU will not accept any packet received without a valid checksum. See <i>Section 1.1.1 - Computing the CRC-16 Checksum</i> for a brief description of the implementation of the CRC-16 algorithm used by the IMU.
Size+6	CRC16_1	

1.1.1 Computing the CRC-16 Checksum

The IMU uses a modified version of the CRC-16 algorithm to create a checksum based on each 4-bit nibble, rather than the full 8-bit byte. This allows for an efficient lookup-based algorithm, while simultaneously reducing the size of the lookup table from 512 bytes to 32 as compared to a more traditional lookup algorithm. The CRC calculation is based on a polynomial of **0x8005**, and is a reflected implementation.

Example 1 - A lightweight, embedded CRC-16 implementation.

```
// Table for cached 4-bit CRC-16 lookups
static const UInt16 crctable[16] = {
    0x0000, 0xCC01, 0xD801, 0x1400, 0xF001, 0x3C00, 0x2800, 0xE401,
    0xA001, 0x6C00, 0x7800, 0xB401, 0x5000, 0x9C01, 0x8801, 0x4400 };

/*! Calculates a 16-bit cyclic redundancy check on blocks of 8-bit data of
 * arbitrary length.
 * \param pBuf points to a buffer of data.
 * \param len is the size of the buffer in bytes.
 * \return The 16-bit CRC checksum of the buffer. */
UInt16 CRC16(const UInt8 *pData, UInt32 len)
{
    UInt16 crc = 0;           // CRC-16 accumulator, initial value 0x0000

    // While there are more bytes to encode
    while (len--)
    {
        // CRC the lower 4 bits
        crc = (crc >> 4) ^ crctable[((crc ^ (*pData & 0xF)) & 0xF)];

        // Now run the upper 4 bits
        crc = (crc >> 4) ^ crctable[((crc ^ (*pData >> 4)) & 0xF)];

        // Move on to the next element
        pData++;
    }

    // Return the cumulative CRC-16 value
    return crc;
}

} // CRC16
```

2.2 High Speed Serial Packet Format

The Crista IMU also supports another serial interface that groups similar telemetry data together, reducing the amount of packet overhead relative to the amount of telemetry data and therefore allowing it to be output at a much higher rate.

The main difference between a high speed serial packet and a standard serial packet is that the data length is not limited to a range of 1 to 8.

Note: High speed packets of type *HS_RAW_IMU_MSG* do not contain a length byte. See Section 4.27 - *HS_RAW_IMU_MSG - High Speed Raw IMU Data* for more information.

3 CAN Interface

CAN is a multi-drop 2-wire differential serial bus, typically used for automotive applications. The CAN protocol provides a well defined datalink layer that includes frame identification and validation. A CAN frame can have from 0 to 8 bytes of data. The CAN interface used in the IMU is run at 1 Mbps and is based on the CAN 2.0B specification, meaning that all CAN frames transmitted to and from the IMU have a 29-bit identifier associated with them (**Table 2**).

3.1 CAN Message ID (CANnelli 2.0)

The 29-bit CAN frame identifier can be interpreted as follows:

Table 2 - CANnelli 2.0 Message Identifier Scheme

Bit	Group	Meaning
0	5-bit message group ID from 0 to 31	The group ID identifies the category of device that this CAN came from or goes to. The group ID for all IMU messages—in both directions—is 1.
1		
2		
3		
4		
5	8-bit message type from 0 to 255	The frame type identifies the contents of the CAN frame, identical to the packet type identifier used by the serial interface. See Table 3 for a complete list of frame types. Note that all identifiers beginning with HS_ correspond to serial-only packet types.
6		
7		
8		
9		
10		
11		
12		
13	16-bit serial number from 1 to 65535	The serial number of the IMU either receiving or sending this frame. All IMUs on the bus will ignore messages not directed to their serial numbers. Also note that 0x0000 is a special serial number reserved to indicate a broadcast frame that all IMUs present on the bus should receive and process.
14		
15		
16		
17		
18		
19		
20		
21		
22		
23		
24		
25		
26		
27		
28		

4 CAN Packet Format

As specified by the CAN 2.0B standard, all CAN packets sent to and from the IMU must contain the following:

- a) An arbitration field containing the message ID as defined above
- b) A control field containing the length of the message
- c) A data field containing 0 to 8 bytes of payload data
- d) A CRC field containing a 15-bit CRC checksum
- e) An acknowledgement field for confirming packet receipt

4.1 Packet Types

The Crista IMU currently utilizes a wide range of packet types to support the various possible modes of operation. As noted previously, each of these has its own unique packet type identifier (**Table 3**) to ease packet parsing.

Table 3 - Packet Type Identifiers

ID	Name	Dir	Meaning
0	RAWGYRO_IMU_MSG	Out	Gyro readings in 16-bit A/D counts
1	RESERVED		
2	RAWACCEL_IMU_MSG	Out	Accelerometer readings in A/D counts
3	TIMING_IMU_MSG	Out	GPS pulse-per-second timing data
4	RESOLUTION_IMU_MSG	Out	Gyro and accelerometer converted data resolution per bit
5	RESUNITS_GYRO_IMU_MSG	Out	Converted gyro readings
6	RESUNITS_ACCEL_IMU_MSG	Out	Converted accelerometer readings
7	SET_SETTINGS_IMU_MSG	In	Sets the IMU output settings
8	SETTINGS_IMU_MSG	Out	Current IMU output settings
9	MFRCLDATE_IMU_MSG	Out	Contains the dates of assembly and calibration for this IMU
10	SERIALNUMCONFIG_IMU_MSG	Out	Serial number and hardware configuration data
11	SWVERSION_IMU_MSG	Out	Version data from the IMU firmware
12	RESERVED		
13	REQ_CONFIG_IMU_MSG	In	Requests all configuration and settings data from the IMU
14	RESERVED		
15	RESERVED		
16	RESERVED		
17	RESERVED		
18	RESERVED		
19	RESERVED		
20	RESERVED		
21	REQ_CALPARAM_IMU_MSG	In	Requests a calibration parameter value
22	CALPARAM_IMU_MSG	Out	Contains a calibration parameter value
23	RAWGYROTEMPX_IMU_MSG	Out	X gyro temperature reading, in volts
24	RAWGYROTEMPY_IMU_MSG	Out	Y gyro temperature reading, in volts
25	RAWGYROTEMPZ_IMU_MSG	Out	Z gyro temperature reading, in volts
26	RESERVED		
27	CRC_STATUS_IMU_MSG	Out	Sensor head EEPROM CRC status message
28	RESERVED		
29	RESERVED		

30	GYRO_STDEV_IMU_MSG	Out	Gyro oversample noise
31	ACCEL_STDEV_IMU_MSG	Out	Accelerometer oversample noise
32	GYRO_MINIMUM_IMU_MSG	Out	Gyro minimum oversample values
33	ACCEL_MINIMUM_IMU_MSG	Out	Accelerometer minimum oversample values
34	GYRO_MAXIMUM_IMU_MSG	Out	Gyro maximum oversample values
35	ACCEL_MAXIMUM_IMU_MSG	Out	Accelerometer maximum oversample values
253	HS_RAWGYROTEMP_IMU_MSG	Out	High speed serial gyro temperature data
254	HS_RAW_IMU_MSG	Out	High speed serial raw sensor data
255	HS_SERIAL_IMU_MSG	Out	High speed serial converted sensor data

4.2 RAWGYRO_IMU_MSG - Raw Gyro Readings

The raw gyro data packet is sent once every output cycle. It contains the 16-bit A/D count codes for each the three gyroscopes, plus a sequence number common to all telemetry packets sent during the current output cycle. The sequence number is incremented once every output cycle, meaning that any gaps in sequentiality indicate dropped telemetry data.

Table 4 - Raw Gyro Data Message Format

Byte	Name	Meaning
0	GyroX_0	16-bit unsigned number of A/D counts registered on the X gyro analog output line.
1	GyroX_1	
2	GyroY_0	16-bit unsigned number of A/D counts registered on the Y gyro analog output line.
3	GyroY_1	
4	GyroZ_0	16-bit unsigned number of A/D counts registered on the Z gyro analog output line.
5	GyroZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.3 RAWACCEL_IMU_MSG - Raw Accelerometer Readings

The raw accelerometer data packet is sent once every output cycle and contains the 16-bit A/D count codes for each the three accelerometers, plus a sequence number common to all telemetry packets sent during the current output cycle. The sequence number is incremented once every output cycle, meaning that any gaps in sequentiality indicate dropped telemetry data.

Table 5 - Raw Accelerometer Data Message Format

Byte	Name	Meaning
0	GyroX_0	16-bit unsigned number of A/D counts registered on the X accelerometer analog output line.
1	GyroX_1	
2	GyroY_0	16-bit unsigned number of A/D counts registered on the Y accelerometer analog output line.
3	GyroY_1	
4	GyroZ_0	16-bit unsigned number of A/D counts registered on the Z accelerometer analog output line.
5	GyroZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.4 TIMING_IMU_MSG - PPS Timing Data

A GPS pulse-per-second timing packet is sent out on every output cycle along with the sensor telemetry data and is intended to aid in accurately aligning GPS data with IMU telemetry in time. The data consists of the time since the last GPS pulse, the number of pulses received, and a metric for measuring the differences between the IMU and GPS clocks.

This packet also contains a sequence number common to all telemetry packets sent during the current output cycle. The sequence number is incremented once every output cycle, meaning that any gaps in sequentiality indicate dropped telemetry data.

Table 6 - GPS Pulse-Per-Second Timing Data Message Format

Byte	Name	Meaning
0	TimeSincePPS_0	32-bit unsigned number of 10 MHz IMU clock ticks since the last received pulse.
1	TimeSincePPS_1	
2	TimeSincePPS_2	
3	TimeSincePPS_3	
4	Count	8-bit unsigned number of pulses recorded.
5	Sequence	8-bit unsigned sequence number for this output cycle.
6	ClockError_0	16-bit signed difference between the expected and actual number of IMU clock ticks between pulses.
7	ClockError_1	

The clock error, as specified in this packet, is derived in **Equation 1**. T_p represents the timestamp (in IMU clock ticks) upon receipt of pulse p . F_{osc} is the IMU clock frequency (10 MHz).

Equation 1 - PPS clock error calculation.

$$e_{clock} = (T_i - T_{i-1}) - F_{osc} = (T_i - T_{i-1}) - 10^6$$

Note: The time elapsed since the last pulse as recorded in this packet represents the amount of time elapsed between the last pulse and the end of the oversampling period.

4.5 RESOLUTION_IMU_MSG - Engineering Units Output Resolution

This packet is sent in response to a request for IMU settings (see Section 4.13 - REQ_CONFIG_IMU_MSG - Request IMU Settings). It contains the magnitude of the maximum reading the sensor is capable of in either g's or degrees per second, depending on the sensor. For example, a $\pm 300^\circ/s$ gyro range is represented in this packet as 300 and a $\pm 10g$ accelerometer range would be represented as 10. The data in this packet is used to convert the data contained in the engineering units output data to usable floating point values. This is accomplished by calculating a conversion factor based on the values shown in Equation 2. CF is the conversion factor and m is the range number as read from this packet. These values are multiplied by the signed 16-bit integer data from an engineering units data packet, which results in usable engineering units data.

Table 7 - Converted Output Data Resolution Message Format

Byte	Name	Meaning
0	GyroRange_0	32-bit floating point representing the magnitude of the maximum gyro reading in degrees per second.
1	GyroRange_1	
2	GyroRange_2	
3	GyroRange_3	
4	AccelRange_0	32-bit floating point representing the magnitude of the maximum accelerometer reading in gs.
5	AccelRange_1	
6	AccelRange_2	
7	AccelRange_3	

The units of the gyro and accelerometer conversion factors are degrees per second per LSB and meters per second-squared per LSB, respectively. This allows them to be directly multiplied with the converted data received from the IMU.

Equation 2 - Calculating the engineering unit data conversion factor.

$$CF_{gyro} = \frac{2m}{2^{16}}$$

$$CF_{accel} = \frac{19.61m}{2^{16}}$$

Note: This message is only present in IMU firmware versions beginning with 1.1.7. In prior software versions, the sensor ranges are hard-coded to be the same as the range specified upon purchase of the sensor.

4.6 RESUNITS_GYRO_IMU_MSG - Converted Gyro Data

If converted data output is currently enabled (see *Section 4.9 - SETTINGS_IMU_MSG - IMU Output Settings Data*), this packet is sent once every output cycle over the CAN link only. The data it contains represent the converted and corrected gyro readings, which are scaled to fit in a signed 16-bit integer for transmission efficiency. In order to convert from the integer data contained in this packet to engineering units, a conversion factor must first be derived from the sensor ranges as described in *Section 4.5 - RESOLUTION_IMU_MSG - Engineering Units Output Resolution*. A value of **0x7fff** indicates the maximum positive gyro reading. A value of **0x8000** indicates the maximum negative gyro reading.

Note: This packet is never transmitted over the serial link. For more information, see Section 4.28 - HS_SERIAL_IMU_MSG - High Speed Converted IMU Data.

Table 8 - Converted Gyro Data Message Format

Byte	Name	Meaning
0	GyroX_0	16-bit signed X gyro reading in resolution units.
1	GyroX_1	
2	GyroY_0	16-bit signed Y gyro reading in resolution units.
3	GyroY_1	

4	GyroZ_0	16-bit signed Z gyro reading in resolution units.
5	GyroZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.7 RESUNITS_ACCEL_IMU_MSG - Converted Accelerometer Data

If converted data output is currently enabled (see *Section 4.9 - SETTINGS_IMU_MSG - IMU Output Settings Data*), this packet will be sent once every output cycle over the CAN link only. The data it contains represent the converted and corrected accelerometer readings, which are scaled to fit in a signed 16-bit integer for transmission efficiency. In order to convert from the integer data contained in this packet to engineering units, a conversion factor must first be derived from the sensor ranges as described in *Section 4.5 - RESOLUTION_IMU_MSG - Engineering Units Output Resolution*. A value of **0x7fff** indicates the maximum positive accelerometer reading. A value of **0x8000** indicates the maximum negative accelerometer reading.

Note: *This packet is never transmitted over the serial link. For more information, see Section 4.28 - HS_SERIAL_IMU_MSG - High Speed Converted IMU Data.*

Table 9 - Converted Accelerometer Data Message Format

Byte	Name	Meaning
0	AccelX_0	16-bit signed X accelerometer reading in resolution units.
1	AccelX_1	
2	AccelY_0	16-bit signed X accelerometer reading in resolution units.
3	AccelY_1	
4	AccelZ_0	16-bit signed X accelerometer reading in resolution units.
5	AccelZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

Note: *Beginning with firmware version 1.3.0, these packets will contain data that have already been pre-corrected for orthogonality and acceleration bias errors (see Section 5.3 - Cross-Axis and Acceleration Bias Corrections).*

4.8 SET_SETTINGS_IMU_MSG - Configure IMU Output Settings

This message is used to configure the output settings, including the data format, the communications protocol(s) to be used and the sampling and output rates. When the IMU receives this packet, it applies the settings internally and echoes back the received settings in a SETTINGS_IMU_MSG packet (see *Section 4.9 - SETTINGS_IMU_MSG - IMU Output Settings Data*). Failure to receive this message in a reasonable amount of time indicates that the message was not received by the IMU, and should be re-sent.

Note: *Setting the output mode to high-speed raw serial data overrides all other output modes.*

Table 10 - IMU Output Settings Configuration Message Format

Byte	Name	Meaning	
0	OutputTarget	Bit	Meaning
		0-5	Reserved, set to 0.

		6	Set to output data over the CAN bus
		7	Set to output data over the serial port.
1	OutputMode	Bit	Meaning
		0-4	Reserved, set to 0.
		5	Set to output high-speed raw serial data ONLY.
		6	Set to output raw A/D count data.
		7	Set to output engineering units data.
2	Oversample_0	16-bit unsigned number of oversamples per output cycle – must be at least 1.	
3	Oversample_1		
4	OutputPeriod_0	32-bit unsigned sensor telemetry packet output period in microseconds.	
5	OutputPeriod_1		
6	OutputPeriod_2		
7	OutputPeriod_3		

4.9 SETTINGS_IMU_MSG - IMU Output Settings Data

This packet is sent from the IMU under the following conditions:

- As an acknowledgement of a change in IMU output settings (see *Section 4.8 SET_SETTINGS_IMU_MSG - Configure IMU Output Settings*).
- As a response to a request for all IMU settings data as described in *Section 4.13 - REQ_CONFIG_IMU_MSG - Request IMU Settings*.

The packet data contains the current output settings (including the data format), the communications protocol(s) to be used, and the sampling and output rates.

Table 11 - IMU Output Settings Configuration Message Format

Byte	Name	Meaning	
0	OutputTarget	Bit	Meaning
		0-5	Reserved, set to 0.
		6	Set if output is being sent over the CAN bus
		7	Set if output is being sent over the serial port.
1	OutputMode	Bit	Meaning
		0-4	Reserved, set to 0.
		5	Set if outputting high-speed raw serial data ONLY.
		6	Set if outputting raw A/D count data.
		7	Set if outputting engineering units data.
2	Oversample_0	16-bit unsigned number of oversamples per output cycle – must be at least 1.	
3	Oversample_1		
4	OutputPeriod_0	32-bit unsigned sensor telemetry packet output period in microseconds.	
5	OutputPeriod_1		
6	OutputPeriod_2		
7	OutputPeriod_3		

4.10 MFRCALDATE_IMU_MSG - Manufacture and Calibration Dates

This packet is only sent upon request for the IMU system settings (see *Section 4.13 - REQ_CONFIG_IMU_MSG - Request IMU Settings*). It contains the dates that the unit was manufactured and calibrated. This data should not change over the life of the IMU. These values are programmed in only when the unit is first assembled, and when it is initially calibrated.

Table 12 - Manufacture and Calibration Dates Message Format

Byte	Name	Meaning
0	MfrMonth	8-bit unsigned month of manufacture, from 1 to 12.
1	MfrDay	8-bit unsigned day of manufacture, from 1 to 31.
2	MfrYear_0	16-bit unsigned year of manufacture.
3	MfrYear_1	
4	CalMonth	8-bit unsigned month of calibration, from 1 to 12.
5	CalDay	8-bit unsigned day of calibration, from 1 to 31.
6	CalYear_0	16-bit unsigned year of manufacture
7	CalYear_1	

4.11 SERIALNUMCONFIG_IMU_MSG - Hardware Configuration Data

This packet is only sent upon request for the IMU system settings (see *Section 4.13 - REQ_CONFIG_IMU_MSG - Request IMU Settings*). It contains the unit's hardware configuration information. It also includes the serial number, the EEPROM format version number, and the hardware revision of the sensor head, the sensor configuration, and a group of flags indicating minor hardware configuration options.

Table 13 - Hardware Configuration Message Format

Byte	Name	Meaning	
0	SerialNum_0	16-bit unsigned serial number of this IMU, from 1 to 65535.	
1	SerialNum_1		
2	EepromVer	8-bit unsigned EEPROM format version identifier.	
3	HWVerMajor	8-bit unsigned hardware revision major version.	
4	HWVerMinor	8-bit unsigned hardware revision minor version.	
5	AccelConfig	8-bit unsigned accelerometer configuration ID.	
6	GyroConfig	8-bit unsigned gyro configuration ID.	
7	ConfigBits	Bit	Meaning
		0	Set if configured to output data statistics.
		1	Set if the sensor head was calibrated on Piccolo hardware.
		2	Reserved.
		3	Set if accelerometer orthogonality data is valid and available.
		4	Set if gyro acceleration bias data is valid and available.
		5	Set if gyro orthogonality data is valid and available.
		6	Reserved.
7	Set if using 16-bit temperature sampling, clear if using 10-bit sampling.		

The sensor configuration ID numbers are used to indicate what acceleration and angular rate sensors are installed on the sensor head. They also include the ranges they have been configured for (if applicable) The values of these identifiers can be interpreted as indicated in the tables below.

Table 14 - Gyro Configuration Identifiers

ID	Gyro Part Number	Range	Notes
0	Analog Devices ADXRS300	±300°/sec	Prior generation standard gyro
1	Analog Devices ADXRS150	±150°/sec	Special order only
2	Analog Devices ADXRS300	±600°/sec	Special order only
3	Analog Devices ADXRS300	±900°/sec	Special order only
4	RESERVED		
5	Analog Devices ADXRS610	±300°/sec	Current generation standard gyro

Table 15 - Accelerometer Configuration Identifiers

ID	Accelerometer Part No.	Range	Notes
0	Analog Devices ADXL210E	±10g	Prior generation standard accelerometer
1	RESERVED		
2	Kionix KXPB5	±6g	Special order only
3	Kionix KXPB5	±10g	Current generation standard accelerometer

4.12 SWVERSION_IMU_MSG - Software Version Information

This packet is only sent upon request for the IMU system settings (see *Section 4.13 - REQ_CONFIG_IMU_MSG - Request IMU Settings*). It contains the unit’s software version information.

Table 16 - Software Version Information Message Format

Byte	Name	Meaning
0	MajorVersion	8-bit unsigned major software version number.
1	MinorVersion	8-bit unsigned minor software version number.
2	SubVersion	8-bit unsigned software sub-version number
3	Released	Bit Meaning
		0 Set if software is released, otherwise software is a test version.
		1-6 6-bit unsigned patch number for bug fixes.
7	Set if this IMU is using the ‘Enhanced’ processor version, otherwise it is using the ‘Classic’ processor.	
4	VersionMonth	8-bit unsigned month of the software release; 1-12.
5	VersionDay	8-bit unsigned day of the software release; 1-31.
6	VersionYear_0	16-bit unsigned year of the software release.
7	VersionYear_1	

4.13 REQ_CONFIG_IMU_MSG - Request IMU Settings

This message is sent to the IMU as a request for all of the major configuration data. Typically the payload data of this packet will be empty (i.e. length zero), but any incidental payload data that it may contain will be ignored. Upon receipt, the IMU immediately responds with the following five configuration packets:

- RESOLUTION_IMU_MSG (see *Section 4.5*)
- SETTINGS_IMU_MSG (see *Section 4.9*)
- MFRCALDATE_IMU_MSG (see *Section 4.10*)
- SERIALNUMCONFIG_IMU_MSG (see *Section 4.11*)
- SWVERSION_IMU_MSG (see *Section 4.12*)
- CRC_STATUS_IMU_MSG (see *Section 4.19*)

The system settings contained in these packets, with the possible exception of the output settings data and the CRC status information, is constant throughout normal operation. Requesting this data is only done once at system start-up (if at all) depending on the mode of operation.

Table 17 - IMU Settings Request Message Format

Byte	Name	Meaning
This message contains no data bytes		

4.14 REQ_CALPARAM_IMU_MSG - Request Calibration Parameter

To request a calibration parameter from the IMU, a REQ_CALPARAM_IMU_MSG must be sent containing the identifying number of the parameter to read from EEPROM. A more in-depth look at calibration parameters and their ID numbers can be found in *Section 6.1 - Sensor Calibration Parameters*.

Table 18 - Calibration Parameter Request Message Format

Byte	Name	Meaning
0	ParamNumber	8-bit unsigned parameter number to be read from EEPROM.

4.15 CALPARAM_IMU_MSG - Calibration Parameter Data

This message is sent in response to a request for a calibration parameter from the IMU as described in *Section 4.14 - REQ_CALPARAM_IMU_MSG - Request Calibration Parameter*. The contents of the message are the parameter's ID number, and the 32-bit floating point value corresponding to that calibration parameter (as loaded into the IMU EEPROM memory).

Table 19 - Calibration Parameter Message Format

Byte	Name	Meaning
0	ParamNumber	8-bit unsigned parameter number contained in this message.

1	CalParam_0	32-bit floating point value of the parameter as read from EEPROM.
2	CalParam_1	
3	CalParam_2	
4	CalParam_3	

4.16 RAWGYROTEMPX_IMU_MSG - X Gyro Temperature

Every one-and-a-half seconds a message containing the average temperature, measured by one of the three on-board gyros, is sent by the IMU. This data can be used in the algorithm for temperature compensation of raw data described in *Section 5 - Raw Data*. If a temperature reading is desired in units other than volts, refer to the applicable data sheet for the installed gyro hardware. See *Section 4.11 - SERIALNUMCONFIG_IMU_MSG - Hardware Configuration Data* for more information on determining the sensor head gyro configuration.

Table 20 - X Gyro Temperature Message Format

Byte	Name	Meaning
0	TempX_0	32-bit floating point X gyro temperature reading in volts.
1	TempX_1	
2	TempX_2	
3	TempX_3	

4.17 RAWGYROTEMPY_IMU_MSG - Y Gyro Temperature

Every one-and-a-half seconds, a message containing the average temperature measured by one of the three on-board gyros is sent by the IMU. This data can then be used in the algorithm for temperature compensation of raw data described in *Section 5 - Raw Data*. If a temperature reading is desired in units other than volts, refer to the applicable data sheet for the installed gyro hardware. See *Section 4.11 - SERIALNUMCONFIG_IMU_MSG - Hardware Configuration Data* for more information on determining the sensor head gyro configuration.

Table 21 - Y Gyro Temperature Message Format

Byte	Name	Meaning
0	TempY_0	32-bit floating point Y gyro temperature reading in volts.
1	TempY_1	
2	TempY_2	
3	TempY_3	

4.18 RAWGYROTEMPZ_IMU_MSG - Z Gyro Temperature

Every one-and-a-half seconds, a message containing the average temperature measured by one of the three on-board gyros is sent by the IMU. This data can then be used in the algorithm for temperature compensation of raw data described in *Section 5 - Raw Data*. If a temperature reading is desired in units other than volts, refer to the applicable data sheet for the installed gyro hardware. See *Section 4.11 - SERIALNUMCONFIG_IMU_MSG - Hardware Configuration Data* for more information on determining the sensor head gyro configuration.

Table 22 - Z Gyro Temperature Message Format

Byte	Name	Meaning
0	TempZ_0	32-bit floating point Z gyro temperature reading in volts.
1	TempZ_1	
2	TempZ_2	
3	TempZ_3	

4.19 CRC_STATUS_IMU_MSG - Sensor Dead Data Integrity Status

This packet is only sent upon request for the IMU system settings (see *Section 4.13 - REQ_CONFIG_IMU_MSG - Request IMU Settings*). It contains a single byte that indicates the integrity of the data stored in the system’s EEPROM memory.

Each time data is written to the EEPROM, a CRC-16 checksum is calculated and stored into memory. This allows the system to internally calculate the EEPROM checksum and compare it to the value stored in memory. If the two values do not match, then there is a data integrity error and the data that the EEPROM chip contains may no longer be valid.

When this packet is requested, the IMU performs that check and returns a byte indicating the result of the comparison.

- **0x00** indicates a valid checksum
- **0xFF** indicates that the checksums do not match

Table 23 - Sensor Head CRC Status Message Format

Byte	Name	Meaning
0	CrcStatus	0x00 – CRC of the EEPROM data matches the stored version, or 0xFF - The CRC values do not match.

4.20 GYRO_STDEV_IMU_MSG - Gyro Oversample Noise

If the IMU is configured to output data statistics and to record more than one oversample per output period, it will internally track an estimate of the standard deviation of the oversamples using the method defined in **Equation 3**. When the ensuing telemetry data output cycle occurs, the standard deviation estimator outputs its current values and resets itself. This packet contains an estimate of the amount of noise recorded during the sample period from the three gyros.

Table 24 - Gyro Oversample Noise Message Format

Byte	Name	Meaning
0	GyroStDevX_0	16-bit unsigned standard deviation of the X gyro readings registered in this sample period, in A/D counts.
1	GyroStDevX_1	
2	GyroStDevY_0	16-bit unsigned standard deviation of the Y gyro readings registered in this sample period, in A/D counts.
3	GyroStDevY_1	
4	GyroStDevZ_0	16-bit unsigned standard deviation of the Z gyro readings registered in this sample period, in A/D counts.
5	GyroStDevZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.21 ACCEL_STDEV_IMU_MSG - Accelerometer Oversample Noise

If the IMU is configured to output data statistics and to record more than one oversample per output period, it will internally track an estimate of the standard deviation of the oversamples using the method in **Equation 3**. When the ensuing telemetry data output cycle occurs, the standard deviation estimator outputs its current values and resets itself. This packet contains an estimate of the amount of noise recorded during the sample period from the three accelerometers.

Table 25 - Accelerometer Oversample Noise Message Format

Byte	Name	Meaning
0	AccelStDevX_0	16-bit unsigned standard deviation of the X accelerometer readings registered in this sample period, in A/D counts.
1	AccelStDevX_1	
2	AccelStDevY_0	16-bit unsigned standard deviation of the Y accelerometer readings registered in this sample period, in A/D counts.
3	AccelStDevY_1	
4	AccelStDevZ_0	16-bit unsigned standard deviation of the Z accelerometer readings registered in this sample period, in A/D counts.
5	AccelStDevZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

Equation 3 - Online algorithm for estimating sample standard deviations.

$$\hat{\sigma} = \sqrt{\frac{1}{N} \sum_{i=1}^N \left(x_i - \frac{1}{i} \sum_{j=1}^i x_j \right)^2}$$

4.22 GYRO_MINIMUM_IMU_MSG - Gyro Minimum Value

If the IMU is configured to output data statistics and to record more than one oversample per output period, it will internally track the minimum values read from the analog-to-digital converter over the entire span of the output period. When the ensuing telemetry data output cycle occurs, the minimum value counter sends its current value and resets itself. This packet contains the lowest voltages recorded from the three gyro analog outputs.

Table 26 - Gyro Minimum Oversample Message Format

Byte	Name	Meaning
0	MinGyroX_0	16-bit unsigned minimum X gyro reading registered during the last sample period, in A/D counts.
1	MinGyroX_1	
2	MinGyroY_0	16-bit unsigned minimum Y gyro reading registered during the last sample period, in A/D counts.
3	MinGyroY_1	
4	MinGyroZ_0	16-bit unsigned minimum Z gyro reading registered during the last sample period, in A/D counts.
5	MinGyroZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.23 ACCEL_MINIMUM_IMU_MSG - Accelerometer Minimum Value

If the IMU is configured to output data statistics and to record more than one oversample per output period, it will internally track the minimum values read from the analog-to-digital

converter over the entire span of the output period. When the ensuing telemetry data output cycle occurs, the minimum value counter sends its current value and resets itself. This packet will contain the lowest voltages recorded from the three accelerometer analog outputs.

Table 27 - Accelerometer Minimum Oversample Message Format

Byte	Name	Meaning
0	MinAccelX_0	16-bit unsigned minimum X accelerometer reading registered during the last sample period, in A/D counts.
1	MinAccelX_1	
2	MinAccelY_0	16-bit unsigned minimum Y accelerometer reading registered during the last sample period, in A/D counts.
3	MinAccelY_1	
4	MinAccelZ_0	16-bit unsigned minimum Z accelerometer reading registered during the last sample period, in A/D counts.
5	MinAccelZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.24 GYRO_MAXIMUM_IMU_MSG - Gyro Maximum Value

If the IMU is configured to output data statistics and to record more than one oversample per output period, it will internally track the maximum values read from the analog-to-digital converter over the entire span of the output period. When the ensuing telemetry data output cycle occurs, the maximum value counter outputs its current values and resets itself. This packet will contain the highest voltages recorded from the three gyro analog outputs.

Table 28 - Gyro Maximum Oversample Message Format

Byte	Name	Meaning
0	MaxGyroX_0	16-bit unsigned maximum X gyro reading registered during the last sample period, in A/D counts.
1	MaxGyroX_1	
2	MaxGyroY_0	16-bit unsigned maximum Y gyro reading registered during the last sample period, in A/D counts.
3	MaxGyroY_1	
4	MaxGyroZ_0	16-bit unsigned maximum Z gyro reading registered during the last sample period, in A/D counts.
5	MaxGyroZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.25 ACCEL_MAXIMUM_IMU_MSG - Accelerometer Maximum Value

If the IMU is configured to output data statistics and to record more than one oversample per output period, it will internally track the maximum values read from the analog-to-digital converter over the entire span of the output period. When the ensuing telemetry data output cycle occurs, the maximum value counter outputs its current values and resets itself. This packet will contain the highest voltages recorded from the three accelerometer analog outputs.

Table 29 - Accelerometer Maximum Oversample Message Format

Byte	Name	Meaning
0	MaxAccelX_0	16-bit unsigned maximum X accelerometer reading registered during the last sample period, in A/D counts.
1	MaxAccelX_1	
2	MaxAccelY_0	16-bit unsigned maximum Y accelerometer reading registered during the last sample

3	MaxAccelY_1	period, in A/D counts.
4	MaxAccelZ_0	16-bit unsigned maximum Z accelerometer reading registered during the last sample period, in A/D counts.
5	MaxAccelZ_1	
6	Sequence	8-bit unsigned sequence number for this output cycle.

4.26 HS_RAWGYROTEMP_IMU_MSG - High Speed Gyro Temperature

When the high speed raw serial output mode is selected, this packet will be the only telemetry data that the IMU outputs besides the high speed raw sensor data message described in the next section. This message contains the raw A/D count readings from the three gyro temperature sensors. It also contains a temperature-data-specific sequence number. This number is incremented once every temperature data output cycle. Any gaps in sequentiality indicate missed packets.

Based on the temperature sampling resolution indicated in bit 7 of the configuration bits described in *Section 4.11 SERIALNUMCONFIG_IMU_MSG - Hardware Configuration Data*, a conversion factor is required to convert the values contained in this packet to voltages. The conversion factor that the A/D count values must be multiplied by depends on whether the IMU is configured for 16-bit or 10-bit resolution temperature sampling shown in **Equation 4**.

Table 30 - High Speed Serial Raw Temperature Data Message Format

Byte	Name	Meaning
0	GyroTempX_0	16-bit unsigned number of 10- or 16-bit A/D counts measured on the X gyro temperature line.
1	GyroTempX_1	
2	GyroTempY_0	16-bit unsigned number of 10- or 16-bit A/D counts measured on the Y gyro temperature line.
3	GyroTempY_1	
4	GyroTempZ_0	16-bit unsigned number of 10- or 16-bit A/D counts measured on the Z gyro temperature line.
5	GyroTempZ_1	
6	TempSequence	8-bit unsigned sequence number for this gyro temperature data output cycle.

Equation 4 - Computing raw temperature conversion factors.

$$(1) \quad CF_{10} = \frac{V_{ref}}{2^{10}} = \frac{5}{1024} \approx 4.8828 \times 10^{-3}$$

$$(2) \quad CF_{16} = \frac{V_{ref}}{2^{16}} = \frac{4.096}{65536} = 6.25 \times 10^{-5}$$

4.27 HS_RAW_IMU_MSG - High Speed Raw IMU Data

When the high speed raw serial output mode is selected, this packet will be the only telemetry data that the IMU outputs besides the high speed raw gyro temperature message described in the previous section. This message contains the raw A/D count readings from all six IMU sensors. It also contains a sequence number. This number is incremented once every output cycle. Gaps in sequentiality indicate dropped telemetry data.

Note: For backward-compatibility reasons, this packet does not contain a size byte in its header and is guaranteed to always have 13 bytes of payload data.

Table 31 - High Speed Serial Raw Sensor Data Message Format

Byte	Name	Meaning
0	GyroX_0	16-bit unsigned number of A/D counts registered on the X gyro analog output line.
1	GyroX_1	
2	GyroY_0	16-bit unsigned number of A/D counts registered on the Y gyro analog output line.
3	GyroY_1	
4	GyroZ_0	16-bit unsigned number of A/D counts registered on the Z gyro analog output line.
5	GyroZ_1	
6	AccelX_0	16-bit unsigned number of A/D counts registered on the X accelerometer analog output line.
7	AccelX_1	
8	AccelY_0	16-bit unsigned number of A/D counts registered on the Y accelerometer analog output line.
9	AccelY_1	
10	AccelZ_0	16-bit unsigned number of A/D counts registered on the Z accelerometer analog output line.
11	AccelZ_1	
12	Sequence	8-bit unsigned sequence number for this output cycle.

4.28 HS_SERIAL_IMU_MSG - High Speed Converted IMU Data

In place of sending a series of data packets over the serial link for converted data output (specifically the RESUNITS_XXX_IMU_MSG and TIMING_IMU_MSG packets), the Crista IMU combines all relevant engineering units telemetry into a single packet to reduce the ratio of packet overhead-to-payload data.

This packet is sent once every output cycle when the IMU is in converted data mode. It includes a sequence number that is incremented once every output cycle. Gaps in sequentiality indicate dropped telemetry data. In order to convert from the integer data contained in this packet to engineering units, a conversion factor must first be derived from the sensor ranges as described in *Section 4.5 - RESOLUTION_IMU_MSG - Engineering Units Output Resolution*.

Table 32 - High Speed Converted Data Serial Packet Format

Byte	Name	Meaning
0	GyroX_0	16-bit signed X gyro reading in resolution units.
1	GyroX_1	
2	GyroY_0	16-bit signed Y gyro reading in resolution units.
3	GyroY_1	
4	GyroZ_0	16-bit signed Z gyro reading in resolution units.
5	GyroZ_1	
6	AccelX_0	16-bit signed X accelerometer reading in resolution units.
7	AccelX_1	
8	AccelY_0	16-bit signed Y accelerometer reading in resolution units.
9	AccelY_1	
10	AccelZ_0	16-bit signed Z accelerometer reading in resolution units.
11	AccelZ_1	
12	TimeSincePPS_0	32-bit unsigned number of 10 MHz IMU clock ticks since the last received GPS

13	TimeSincePPS_1	pulse.
14	TimeSincePPS_2	
15	TimeSincePPS_3	
16	PPSCount	8-bit unsigned number of GPS pulses recorded.
17	Sequence	8-bit unsigned sequence number for this output cycle.

5 Raw Data

An optional raw data mode is provided by the Crista IMU in order to give the user the ability to increase data output rates by offloading the bulk of the work to external post-processing. This can be achieved by querying the IMU for sensor calibration data and applying these values to the raw data received from the unit in the manner described below. All parameters in the following equations can be retrieved from the EEPROM as described in *Section 6.1 - Sensor Calibration Parameters*.

5.1 Converting to Engineering Units

The raw data received from the IMU corresponds to the average number of 16-bit analog-to-digital counts registered on each oversample. The first step in the process of converting raw IMU data to usable engineering data is converting the count values into volts. The least significant bit (LSB) in the raw count value is equivalent to a voltage known as the LSB size (shown in **Equation 5**). V_{ref} is the reference voltage being fed into the A/D converter that is fixed at 4.096 volts.

Equation 5 - LSB size for the Crista IMU.

$$LSBSize = \frac{V_{ref}}{2^{16}} = \frac{4.096}{65536} = 6.25 \times 10^{-5} \text{ Volts/LSB}$$

This implies that to convert from A/D counts to volts, the count value is multiplied by the LSB size to arrive at the analog sensor output reading in volts (shown in **Equation 6**).

V is the voltage and C is the raw A/D count value for the sensor.

Equation 6 - Converting A/D counts (C) to volts (V).

$$V = C \cdot LSBSize = C \cdot 6.25 \times 10^{-5}$$

5.2 Applying Calibration Data

Once the raw A/D counts have been successfully converted to analog output voltages (*Section 5.1*), the next step is to apply the offsets and gains. These values have been pre-computed based on a rigorous series of tests performed at CCT before each IMU is shipped.

There are two voltage offsets that need to be applied to the data at this point. Of these, one is constant and the other varies with the current temperature of the part. After these offsets have been applied, a single gain number must then be multiplied by the resultant voltage in order to obtain the data in engineering units (shown in **Equation 7**). X is the sensor reading in engineering units, g is the sensor's gain, O_s is the static offset, and O_t is the voltage offset with respect to temperature.

Equation 7 - Converting volts to engineering units

$$X = g(V + O_s - O_t)$$

In order to calculate the temperature-induced offset (O_t), the pre-loaded sensor calibrations contain 10-point tables of gyro temperatures and the sensor voltage offsets that correspond to those temperatures. The temperature offset for any given sensor can be calculated by linearly interpolating within these two tables using the gyro temperature that corresponds to that sensor (see *Section 1.1.2 - Interpolating Temperature-Induced Accelerometer Offsets*).

Note: The gain numbers for the gyros are a special case. The lines pass through a voltage divider before arriving at the analog-to-digital converter that causes their voltages to be reduced to 80.85% of their actual output values. This means that the gyro gains all need to be divided by 0.8085 before applying it as shown in the above equation.

In **Equation 7**, the gain number for the Y gyro must be multiplied by -1 before using it as the g coefficient.

1.1.2 Interpolating Temperature-Induced Accelerometer Offsets

The interpolation of temperature-induced offsets for accelerometers is based on the temperature reading from the gyro that is physically nearest the sensor. This is because the gyros and accelerometers are in arbitrary locations on the IMU board layouts, and only the gyros are equipped with temperature sensors. The correct temperature channels to use when calculating the offsets for the accelerometers are shown out in the **Table 33**.

Table 33 - Gyro Temperature Channels to Accelerometers

Sensor	Temperature Channel for Sensor Calibration
X Accelerometer	Y gyro temperature
Y Accelerometer	X gyro temperature
Z Accelerometer	Y gyro temperature

5.3 Cross-Axis and Acceleration Bias Corrections

Raw data is represented in engineering units and has been compensated for scale factor differences and temperature-induced errors, however, the sensors remain susceptible to two other major sources of error:

- **Sensor misalignment** - The boards holding the sensors themselves are not entirely orthogonal to one another
- **Acceleration sensitivity** - Increased accelerations exert influence on the output of the gyroscopes

The IMU contains three matrices to combat these errors. Performing various matrix operations with these matrices, and the 3-dimensional sensor output vectors (i.e. X, Y, and Z accelerations), is the final step in converting raw output data into usable numbers.

To mitigate sensor alignment errors, 3x3 sensor orthogonality matrices are provided in the on-board EEPROM. They need to be multiplied by the engineering-units sensor data vector to compensate for any non-orthogonalities (**Equation 8**.) \vec{G}' and \vec{A} are the gyro and accelerometer data vectors, \vec{G}'' and \vec{A}' are the cross-axis corrected gyro and accelerometer data vectors, and GO and AO are the gyro and accelerometer orthogonality matrices. Note that the gyro vector has already been compensated for acceleration sensitivity.

Equation 8 - Sensor orthogonality corrections.

$$\begin{bmatrix} G''_x \\ G''_y \\ G''_z \end{bmatrix} = \begin{bmatrix} GO_{xx} & GO_{xy} & GO_{xz} \\ GO_{yx} & GO_{yy} & GO_{yz} \\ GO_{zx} & GO_{zy} & GO_{zz} \end{bmatrix} \cdot \begin{bmatrix} G'_x \\ G'_y \\ G'_z \end{bmatrix}$$

$$\begin{bmatrix} A'_x \\ A'_y \\ A'_z \end{bmatrix} = \begin{bmatrix} AO_{xx} & AO_{xy} & AO_{xz} \\ AO_{yx} & AO_{yy} & AO_{yz} \\ AO_{zx} & AO_{zy} & AO_{zz} \end{bmatrix} \cdot \begin{bmatrix} A_x \\ A_y \\ A_z \end{bmatrix}$$

Gyro acceleration bias data is stored in the sensor head’s EEPROM in the form of a 3x3 matrix. This matrix containing the offset effects of accelerations in all three dimensions for each of the three gyros in units of volts per meter per second-squared.

To compute the biases, the vector of cross-axis corrected accelerometer readings must be:

- a) Multiplied by the gyro acceleration bias matrix
- b) Multiplied by the vector of gyro gains
- c) Subtracted from the un-corrected gyro vector.

This process is shown below in **Equation 9**, where \vec{G}' is the resultant vector, GAB is the gyro acceleration bias matrix, \vec{g} is the gain vector, and \vec{A}' and \vec{G} are the corrected accelerometer and uncorrected gyro data vectors.

After these calculations have been performed, the data have been fully converted and corrected, and are now in usable engineering units.

Note: Equation 9 assumes that the gyro gain transformations described in the notes at the end of Section 5.1 have been applied to the values in \vec{g}

Equation 9 - Removing acceleration bias effects.

$$(1) \quad \begin{bmatrix} B_x \\ B_y \\ B_z \end{bmatrix} = \begin{bmatrix} GAB_{xx} & GAB_{xy} & GAB_{xz} \\ GAB_{yx} & GAB_{yy} & GAB_{yz} \\ GAB_{zx} & GAB_{zy} & GAB_{zz} \end{bmatrix} \cdot \begin{bmatrix} A'_x \\ A'_y \\ A'_z \end{bmatrix}$$

$$(2) \quad \begin{bmatrix} B'_x \\ B'_y \\ B'_z \end{bmatrix} = \begin{bmatrix} B_x & B_y & B_z \end{bmatrix} \cdot \begin{bmatrix} g_x \\ g_y \\ g_z \end{bmatrix}$$

$$(3) \quad \begin{bmatrix} G'_x \\ G'_y \\ G'_z \end{bmatrix} = \begin{bmatrix} G_x \\ G_y \\ G_z \end{bmatrix} - \begin{bmatrix} B'_x \\ B'_y \\ B'_z \end{bmatrix}$$

Note: Beginning with firmware version 1.3.0, an IMU configured to output engineering units data (see Section 4.9 - SETTINGS_IMU_MSG - IMU Output Settings Data) will perform the calculations in Equation 9 internally, so the only conversion that is necessary is the one described in Section 4.5 - RESOLUTION_IMU_MSG - Engineering Units Output Resolution.

6 Serial EEPROM

The Crista Sensor Head is equipped with a Microchip 25LC640 serial EEPROM chip that comes pre-loaded with factory calibration data, which manages all non-volatile system configuration settings. The 25LC640 uses SPI as its communications interface. The data sheet can be found at [Microchip's](http://www.microchip.com) website.

Note: EEPROM data, like packet payload data, is always stored in big-endian format. This means that the most significant byte (MSB) of any multi-byte data type resides at the lower address.

Table 34 - EEPROM Memory Layout

Addr	Name	Meaning
0	Init_0	First EEPROM initialization byte, should be 0x55.
1	Init_1	Second EEPROM initialization byte, should be 0xAA.
2	CRC_0	16-bit unsigned CRC-16 checksum of all the currently stored data.
3	CRC_1	
4	RESERVED	
5	SerialNumber_0	16-bit unsigned serial number of the Crista Sensor Head.
6	SerialNumber_1	
7	EepromFormat	8-bit unsigned EEPROM format version.
8	HardwareRev	8-bit unsigned Sensor Head hardware revision.
9	Config_0	24-bit configuration field as defined previously in Section 4.11 - SERIALNUMCONFIG_IMU_MSG - Hardware Configuration Data.
10	Config_1	
11	Config_2	
12	MfrMonth	8-bit unsigned month of manufacture, 1-12.
13	MfrDay	8-bit unsigned day of manufacture, 1-31.
14	MfrYear_0	16-bit unsigned year of manufacture.
15	MfrYear_1	
16	CalMonth	8-bit unsigned month of calibration, 1-12.
17	CalDay	8-bit unsigned day of calibration, 1-31.
18	CalYear_0	16-bit unsigned year of calibration.
19	CalYear_1	
...	RESERVED	

404	GyroOrtho	3x3 32-bit floating point gyro ortho matrix.
440	AccelOrtho	3x3 32-bit floating point accelerometer ortho matrix.
476	GyroAccelBias	3x3 32-bit floating point acceleration bias matrix.
512	CalibrationGyroX	128-byte sensor calibration data structures, see Section 6.1 - Sensor Calibration Parameters for the detailed contents of these structures.
640	CalibrationGyroY	
768	CalibrationGyroZ	
896	CalibrationAccelX	
1024	CalibrationAccelY	
1152	CalibrationAccelZ	
1280	RESERVED	
1281	OutputTarget	8-bit output device as defined in Section 4.9.
1282	OutputMode	8-bit output data mode as defined in Section 4.9.
1283	Oversample_0	16-bit unsigned number of oversamples per output cycle.
1284	Oversample_1	
1285	OutputPeriod_0	32-bit unsigned output cycle period in microseconds.
1286	OutputPeriod_1	
1287	OutputPeriod_2	
1288	OutputPeriod_3	
...	RESERVED	
1536	UserData_0	Start of user space.

6.1 Sensor Calibration Parameters

Each sensor has a 128-byte structure in the EEPROM containing the parameters necessary to calibrate its output. This structure is made up of a series of 32, 32-bit floating point parameters in the order shown in **Table 35**. Each hardware configuration has a different method of requesting this data:

When using a bare sensor head, the 4 bytes beginning at the address (**Equation 10**) should be read from the EEPROM. For a Crista IMU assembly, a request for the calibration parameter (**Equation 11**) should be sent to the unit as described in *Section 4.14 - REQ_CALPARAM_IMU_MSG - Request Calibration Parameter*.

Table 35 - Sensor Calibration Parameter Data Structure Format

Parameter	Name	Meaning
0	Gain	The static gain to be applied to this sensor.
1	Offset	The static offset to be applied to this sensor.
2	Temp_0	Reference temperature 0, in volts.
3	Temp_1	Reference temperature 1, in volts.
4	Temp_2	Reference temperature 2, in volts.
5	Temp_3	Reference temperature 3, in volts.
6	Temp_4	Reference temperature 4, in volts.
7	Temp_5	Reference temperature 5, in volts.
8	Temp_6	Reference temperature 6, in volts.

9	Temp_7	Reference temperature 7, in volts.
10	Temp_8	Reference temperature 8, in volts.
11	Temp_9	Reference temperature 9, in volts.
...	RESERVED	
22	TempOffset_0	Voltage offset induced by reference temperature 0.
23	TempOffset_1	Voltage offset induced by reference temperature 1.
24	TempOffset_2	Voltage offset induced by reference temperature 2.
25	TempOffset_3	Voltage offset induced by reference temperature 3.
26	TempOffset_4	Voltage offset induced by reference temperature 4.
27	TempOffset_5	Voltage offset induced by reference temperature 5.
28	TempOffset_6	Voltage offset induced by reference temperature 6.
29	TempOffset_7	Voltage offset induced by reference temperature 7.
30	TempOffset_8	Voltage offset induced by reference temperature 8.
31	TempOffset_9	Voltage offset induced by reference temperature 9.

Equation 10 - Finding the EEPROM address of a calibration parameter.

$$A = S + 4p$$

S is the starting address of the calibration data, and p is the parameter ID from **Table 35**.

Equation 11 - Computing the calibration parameter number to request from the IMU.

$$P = 32s + p + 64,$$

s is the sensor number as described in **Table 36** and p is the parameter number from **Table 35**.

Table 36 - Sensor Identifier Numbers

ID	Sensor	ID	Sensor
0	X gyro	3	X accelerometer
1	Y gyro	4	Y accelerometer
2	Z gyro	5	Z accelerometer