INS330BI Series User Manual

Document Part Number: 7430-1840-01 Preliminary

ACEINNA, Inc.

email: info@aceinna.com, website: www.aceinna.com

WARNING

This product has been developed by ACEINNA exclusively for commercial applications. It has not been tested for, and ACEINNA makes not representation or warranty as to conformance with, any military specifications or that the product is appropriate for any military application or end-use. Additionally, any use of this product for nuclear, chemical, biological weapons, or weapons research, or for any use in missiles, rockets, and/or UAV's of 300km or greater range, or and other activity prohibited by the Export Administration Regulations, is expressly prohibited without the written consent of ACEINNA and without obtaining appropriate US export license(s), when required by US law. Diversion contrary US law is prohibited.

Revision history

Date	Document Revision	Associated FW Version	Description	Author(s)
2021	D.01	10.00.12	Draft	AB / JF /XD

Table of Contents

1. Introduction 8
1.1. Manual overview 8
1.1. Overview of the INS330BI 8
2. Interface 10
2.1. Electrical interface 10
2.1.1. Unit mode configuration. 10
2.1.2. Serial interfaces 10
2.1.3. Synchronization input 11
2.2. Mechanical Drawing & Dimensions 11
2.2.1. Land Pattern 12
2.2.2. Solder Reflow Profile 12
3. Theory of operation 14
3.1. Overview 14
3.2. Device coordinate system 16
3.3. Advanced settings 17
3.4. Data processing chain 17
3.5. User Settings 18
4. UART interface definition 18
4.1. General settings 19
4.2. Number formats 19
4.3. Packet format 20
4.3.1. Packet header 20
4.3.2. Packet type 20
4.3.3. Payload length 20
4.3.4. Payload 20
4.3.5. 16-bit CRC-CCITT 21
4.3.6. Messaging overview 21
5. Standard UART commands and messages 22
5.1. Link test 23
5.1.1. Ping command 23
5.2. Interactive commands 23
5.2.1. Set Master Unit Mode “MM” Packet 23
5.2.2. Get Packet request 23
5.2.3. Error response 24

5.3. Output packets (Polled) 24
5.3.1. Identification data packet 24
5.3.2. Version data packet 25
5.3.3. Test 0 (Detailed BIT and status) packet 25
5.4. Output packets (polled or continuous) 26
5.4.1. INS Solution Packet E3 (default output packet to user) 26
5.4.2. Master Data Packet MG (output from master unit to slave unit) 29
5.4.3. Slave Data Packet “SA” (output from slave to master unit) 30
5.4.4. Scaled sensor data packet 1 (IMU data packet) 30
6. Advanced UART commands 32
6.1. Configuration fields 32
6.1.1. Packet rate divider 33
6.1.2. User UART baudrate 33
6.1.3. Continuous packet type field 33
6.1.4. Digital filter settings 34
6.1.5. Orientation field 34
6.2. Read/Write of configuration fields 37
6.2.1. Write Fields command 37
6.2.2. Write Fields response 37
6.2.3. Set Fields command 38
6.2.4. Set Fields response 39
6.2.5. Read Fields command 39
6.2.6. Read Fields response 40
6.2.7. Get Fields command 40
6.2.8. Get Fields response 41
6.2.9. Entry transportation mode(only applicable for master unit) 41
7. Advanced UART Port BIT 43
7.1. Built In Test (BIT) and Status fields 43
7.2. Master BIT and Status (BITstatus) field 43
7.3. hardwareBIT field 45
7.4. softwareDataBIT field 46
7.5. hardwareStatus field 46
7.6. sensorStatus field 46
8. Bootloader 47

8.1. Updating firmware over UART interface....47
8.1.1. Jump to BootLoader command....47
8.1.2. Write APP command....47
8.1.3. Jump to Application command....48
9. Warranty and support information....49
9.1. Customer service....49
9.2. Contact directory....49
9.3. Return procedure....49
9.3.1. Authorization....49
9.3.2. Identification and protection....49
9.3.3. Sealing the container....50
9.3.4. Marking....50
9.3.5. Warranty....50
Appendix A: Installation and operation of NAV-VIEW....51
Appendix B: CRC calculation for serial port messages....61
Appendix C: Sample Packet Decoding....62

About this Manual

The following symbols are used to provide additional information.

◀ NOTE

Note provides additional information about the topic.

☑ EXAMPLE

This symbol indicates an example that will help the reader understand the terminology.

⚐ IMPORTANT

This symbol defines items that have significant meaning to the user

⚠ WARNING

The user should pay particular attention to this symbol. It means there is a chance that physical harm could happen to either the person or the equipment.

This manual uses the following paragraph heading formats:

1 Heading 1

1.1 Heading 2

1.1.1 Heading 3

1.1.1.1 Heading 4

Normal

1. Introduction

1.1. Manual overview

This manual describes ACEINNA's INS330BI Inertial Measurement Unit (IMU_. The INS330BI will be referred to as "the unit" frequently in this document.

Table 1 below highlights the content in each section and suggests how to use this manual.

Table 1: Manual content

Manual Section	Who Should Read?
Section 1: Manual Overview	All customers should read Sections 1.1 and 1.1.
Section 2: Interface	Customers designing the electrical and mechanical interface to the INS330BI series products should read Section 2.
Section 3: Theory of Operation	All customers should read Section 3.
Section 4-7: UART Port Interface	Customers designing the software Interface To the INS330BI series products UART Port should review Sections 4, 5, 6, 7.
Section 8: INS330BI Bootloader	Customers that intend to update Aceinna firmware in the unit should read Section 8.
Section 9: Warranty and Support Information	All customers should read Section 9.
Appendix A: Installation and Operation of NAV-VIEW	Customers that wish to use NAV-View for evaluation should read Appendix A.
Appendix B: CRC Calculation	All customers that intend to create scripts to interface with serial messages should read Appendix B
Appendix C: Sample Packet Decoding	All customers that intend to create scripts to interface with serial messages should read Appendix C

1.1. Overview of the INS330BI

This manual describes the use of ACEINNA's INS330BI and is intended to be used as a detailed technical reference and operating guide. ACEINNA's INS330BI combines the latest in high-performance commercial MEMS (Micro-electromechanical Systems) sensors and digital signal processing techniques. It can be used as a standalone IMU but it is intended to be used with and external GNSS module to provide a small, cost-effective Inertial Navigation Solution (INS).

The INS330BI is ACEINNA's latest generation of MEMS-based Inertial Systems, building on over a decade of field experience, and encompassing many thousands of deployed units and millions of operational hours in a wide range of land, marine, airborne, and instrumentation applications. It is designed for OEM applications.

At the core of the INS330BI is a triple redundant rugged 6-DOF (Degrees of Freedom) MEMS inertial sensor. The 6-DOF MEMS inertial sensor cluster includes three axes of MEMS angular rate sensing and three axes of MEMS linear acceleration sensing. These sensors are based on rugged, field proven silicon bulk micromachining technology. Each sensor within the cluster is individually factory calibrated for temperature and non-linearity effects during ACEINNA's manufacturing and test process using automated thermal chambers and rate tables.

A key differentiating feature of the INS330BI is the triple redundant 6-DOF MEMS sensor clusters. This redundancy has two direct benefits:

1) Combining multiple sensor readings improves the noise characteristics of the output signal.
2) Using more than one sensor enables the unit to operate through a single sensor-chip failure by detecting the failed part and removing it from the solution. Failures include stuck or railed readings as well as sustained inconsistency between the two sensor sets.

Another unique feature of the INS330BI is the preloaded INS algorithm running an extended Kalman filter to correct for bias drift and error accumulation. When combined with a GNSS receiver it provides the INS solution at 100 Hz rate.

INS330BI has 3 modes of operation:

1. Standalone IMU and GNSS Aided INS Solution (INS requires GNSS receiver to be present)
2. Master GNSS-aided INS solution (reguires Slave INS330BI and GNSS receiver to be present)
3. Slave GNSS-aided steering angle and steering rate information (requires Master INS330BI and GNSS receiver to be present)

The INS330BI IMU is packaged in a lightweight 44 ball BGA surface mount package that is designed for cost-sensitive commercial OEM applications.

ACEINNA's NAV-VIEW 3.X Windows application can be used for communicating with the unit over UART data port. NAV-VIEW 3.X is a powerful Windows-based operating tool that provides field configuration, diagnostics, charting of sensor performance, and data logging with playback.

2. Interface

2.1. Electrical interface

The INS330BI PIN definition is shown in Figure 1.

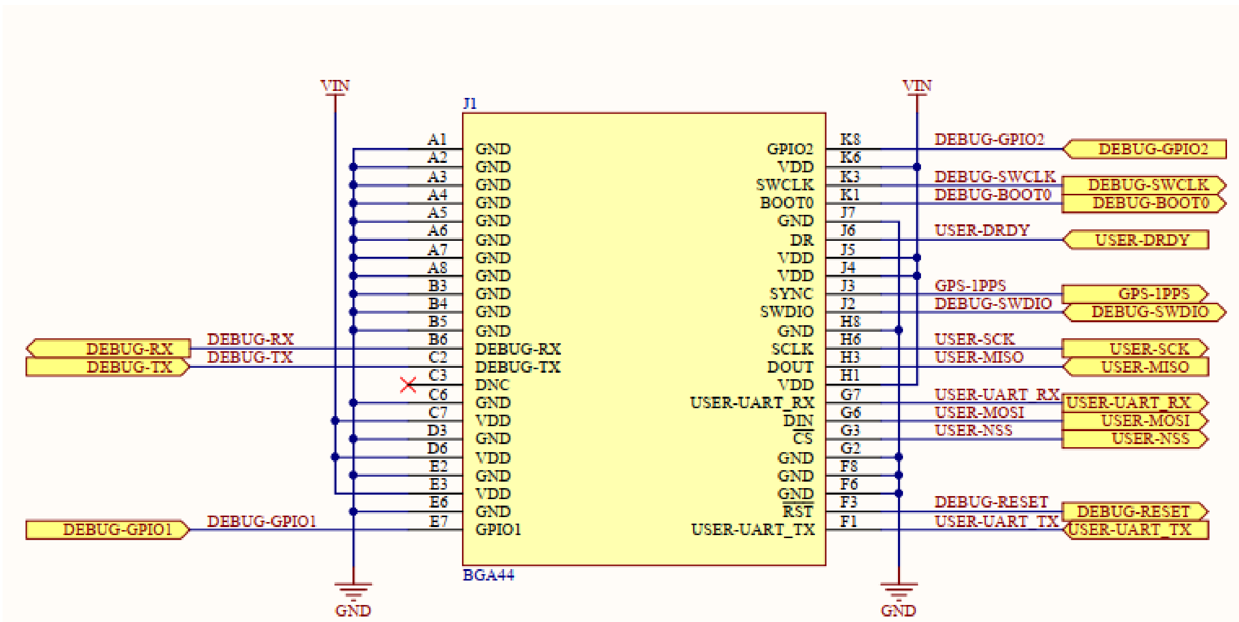


Figure 1 PIN Definition

2.1.1. *Unit mode configuration.*

INS330BI can be configured for work in one of three modes

a. Standalone mode (INS or IMU). Pin DEBUG_GPIO1 needs to be left unconnected.

b. Master mode for dual units solution. Pin DEBUG_GPIO1 tied to VIN.

c. Slave mode for dual unit solution. Pin DEBUG_GPIO1 tied to GND.

2.1.2. *Serial interfaces*

INS330BI unit has two serial interfaces. They used for communication with user system, with GNSS module and between two INS330BI units in dual INS solution.

In single-unit applications USER-UART interface used for communication with host (user) system and DEBUG interface may be left unconnected or used for communication with GNSS module.

In dual-unit applications in Master mode USER-UART interface used for communication with slave unit and DEBUG interface can be shared between host (user) and GNSS module.system and DEBUG interface may be left unconnected or used for communication with GNSS module.

In dual-unit applications in Slave mode USER-UART interface used for communication with master unit and DEBUG interface usually not used.

For the complete UART interface definition, please refer to following Sections 4, 5, 6, 7.

2.1.3. Synchronization input

Pin GPS_1PPS can be used to force synchronization of unit data processing to an external 1 Hz input signal.

2.2. Mechanical Drawing & Dimensions

The INS330BI mechanical drawing and bottom view is defined by the outline drawing in

Figure 2 & Figure 3.

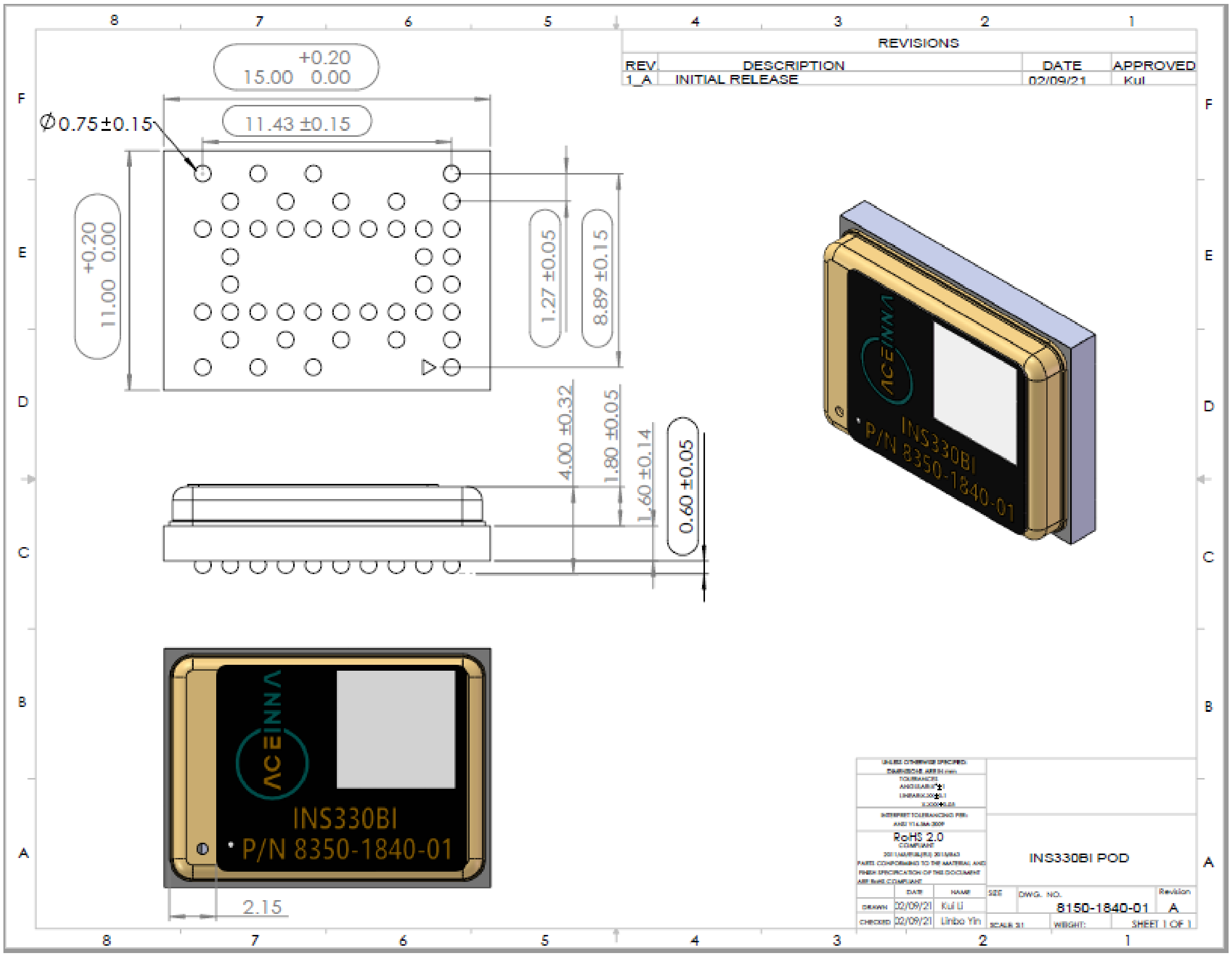


Figure 2: Mechanical Drawing

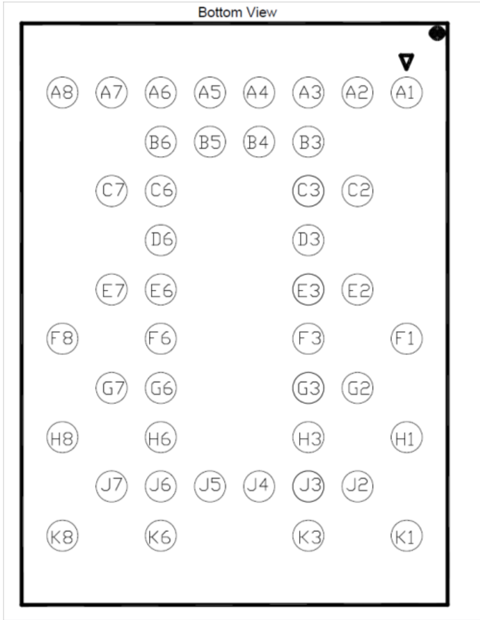


Figure 3 Bottom View

2.2.1. *Land* Pattern

The recommended land pattern of PCB is shown in Figure 4.

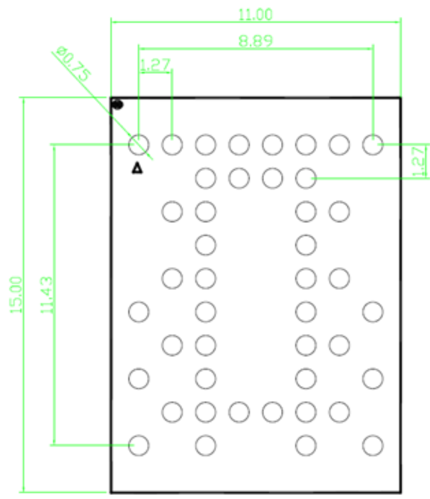


Figure 4: Land Pattern of PCB (units:mm)

2.2.2. *Solder Reflow Profile*

1. BGA ball material is SAC305.
2. The carrier board material of INS330BI is suggested to be Tg180 FR4.
3. Reflow profile for Pb free process.

4. Reflow is limited to 2 times. Second reflow should be applied after device has cooled down to room temperature (25℃).
5. Recommended reflow profile for Pb free process is shown in Figure 5. The time duration of peak temperature (260℃) should be limited to 10 seconds.
6. Type 4 solder paste is recommended for a better SMT quality.
7. Use no clean flux to avoid product contamination by cleaning solvent.
8. It is recommended use underfill glue to manage certain threats to the integrity of the solder joints of the INS330BI, including peeling stress and extended exposure to vibration. Underfill glue is not required for those that do not anticipate exposure to these types of mechanical stresses.
9. Recommended Solder Reflow Profile is shown in the Figure 5.

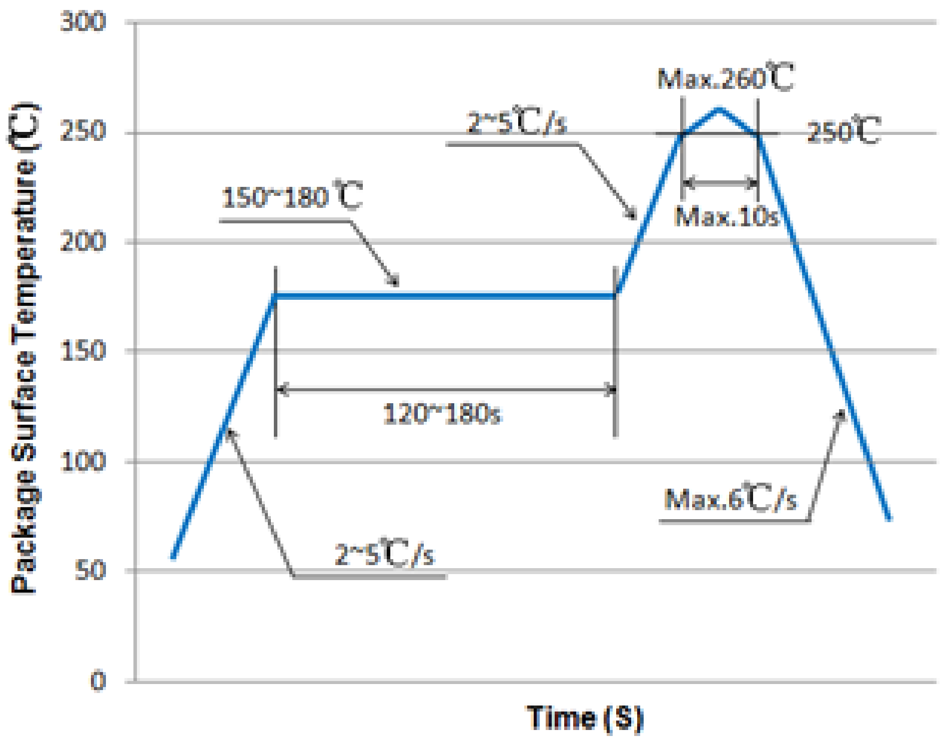


Figure 5: Recommended solder reflow profile

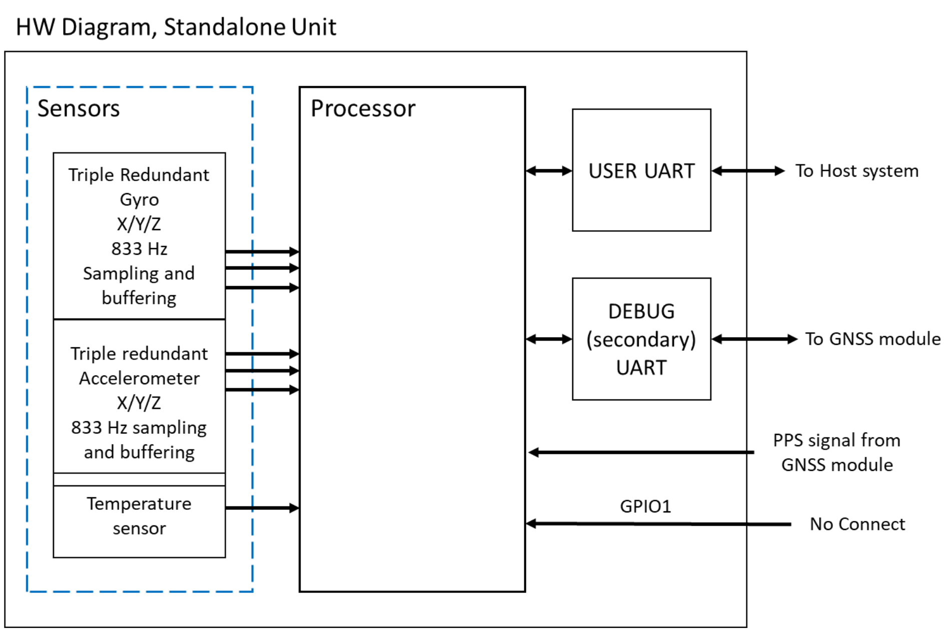
3. Theory of operation

This section of the manual covers detailed theory of operation for the INS330BI.

3.1. Overview

Figure 6 shows the INS330BI hardware block diagrams for the different modes of operation. At the core of the INS330BI are rugged 6-DOF (Degrees of Freedom) MEMS inertial sensors. The 6-DOF MEMS inertial sensors include three axes of MEMS angular rate sensing and three axes of MEMS linear acceleration sensing. These sensors are based on rugged, field proven silicon bulk micromachining technology.

Each sensor is individually factory calibrated using ACEINNA's automated manufacturing process. Sensor errors are compensated for temperature bias, scale factor, non-linearity and misalignment effects using a proprietary algorithm on the data collected during manufacturing. Accelerometer and rate gyro sensor bias shifts over temperature (-40 ˚C to +85 ˚C) are compensated and verified using calibrated thermal chambers and rate tables.



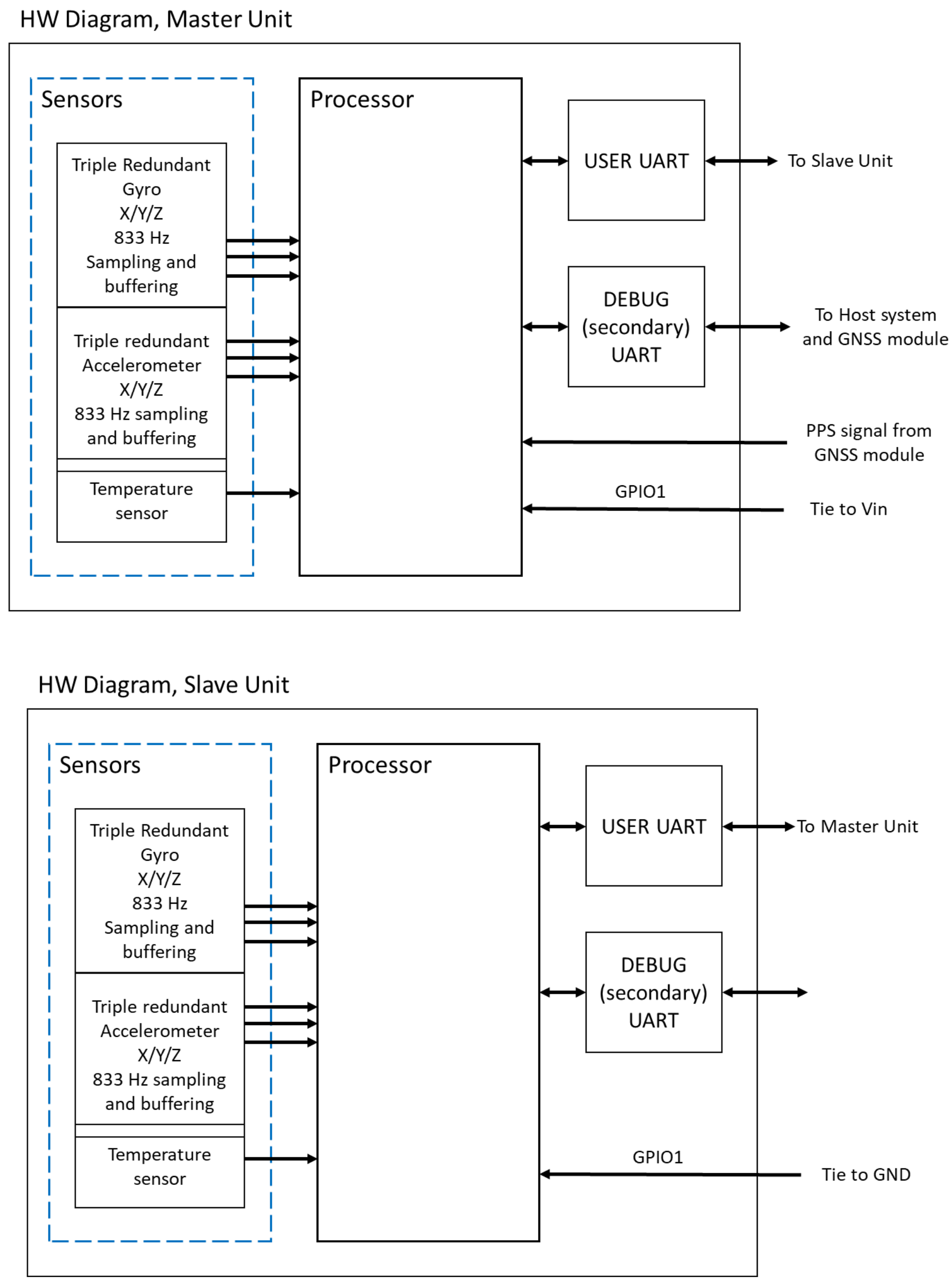


Figure 6: Series hardware block diagrams

The 6-DOF inertial sensor cluster data connected to the processor, which executes data processing algorithms and controls communication with external devices via UART serial interfaces. The UART ports protocol defined in Sections 4, 5, 6.

Figure 7 shows the software block diagram. The 6-DOF data from inertial sensors and aiding devices fed into a high speed 100 Hz signal processing chain. These 6-DOF signals pass through one or more of the processing blocks and are converted into output measurement data as shown. Measurement data packets are available at fixed continuous output rates or on a polled basis.

As shown in the software block diagram, the INS330BI has a unit setting and profile block which configures the unit to user and application specific needs. This feature is one of the more powerful features in the INS330BI architecture as it allows the INS330BI to work in a wide range of commercial applications by settings different modes of operation for the INS330BI IMU.

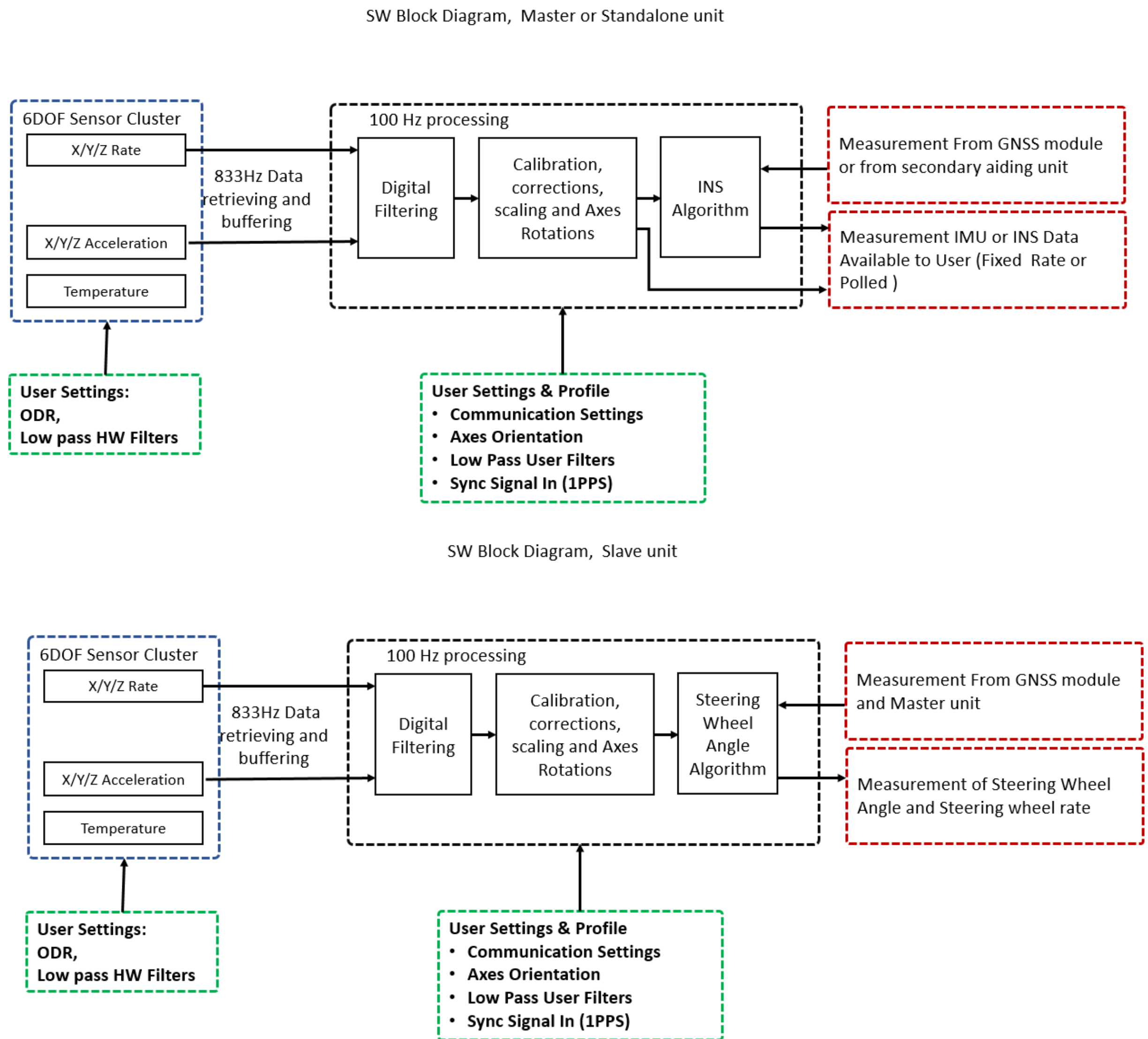


Figure 7: INS330BI Series software block diagrams

3.2. Device coordinate system

The INS330BI Series Inertial System device coordinate frame is shown in Figure 8. As with many elements of the INS330BI, the coordinate system is configurable with either NAV-VIEW or by sending the appropriate serial commands over the UART interface. These configurable elements are known as ***Advanced Settings***. This section of the manual describes the default coordinate system settings of the INS330BI when it leaves the factory.. As with many elements of the INS330BI, the coordinate system is configurable with either NAV-VIEW or by sending the appropriate serial commands over the UART interface. These configurable elements are known as ***Advanced Settings***. This section of the manual describes the default coordinate system settings of the INS330BI when it leaves the factory.

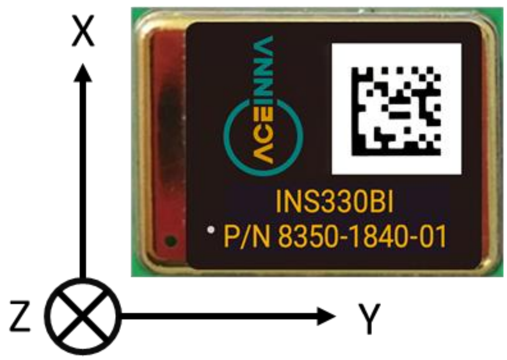


Figure 8: INS330BI device coordinate frame

The axes form an orthogonal SAE right-handed coordinate system. Acceleration is positive when it is oriented towards the positive side of the coordinate axis. For example, an INS330BI sitting on a level table with the default UART orientation setting will measure zero g along the x and y-axes and -1 g along the z-axis. Normal Force acceleration is directed upward, and thus will be reported as negative for the INS330BI UART default orientation setting.

The angular rate sensors are aligned with the same axes. The rate sensors measure angular rotation rate around a given axis. The rate measurements are labeled by the appropriate axis. The direction of a positive rotation is defined by the right-hand rule. With the thumb of your right hand pointing along the axis in a positive direction, your fingers curl around in the positive rotation direction. For example, if the INS330BI is sitting on a level surface and it is rotated counter-clockwise on that surface, the result will be a positive rotation around the z-axis. The x- and y-axis rate sensors would measure zero angular rates, and the z-axis sensor would measure a positive angular rate.

3.3. Advanced settings

The INS330BI IMUs have a number of advanced settings that can be changed. All units support baud rate[1], continuous output packet type, output rate, sensor low pass filtering, and custom axes configuration. The units can be configured using NAV-VIEW, as described in Appendix A, or directly with serial commands as described in following Sections 4, 5, 6, 7. Configuration settings can be made temporary or permanent.

3.4. Data processing chain

Data from sensors cluster (each axis for each sensor separately) passes through programmable low-pass filters, and a signal processing component for sensor error compensation, calibration and axes rotation at rate 100Hz.

The factory calibration data, stored in EEPROM, is used by the processor to remove temperature bias, misalignment, scale factor errors, and non-linearities from the sensor data. Additionally, any advanced

[1] Note: certain combinations of baud-rate, packet-type, and output data rate are invalid because the time to transmit the data exceeds a limit on the permissible message length. The INS330BI limits the output packet width to 80% of the time between data packets. For instance, if the packet is output every 10 milliseconds (100 Hz) then the packet width must be less than 8 milliseconds or the combination is not allowed. This prevents messages from overlapping and causing communication problems. For this reason high baud-rates are suggested.

user settings such as axes rotation are applied to the IMU data. Finally, sensor fault detection is performed on the sensor signals and the individual sensor signals are combined to form a signal with reduced noise characteristics. Then IMU data together with data from external aiding units processed by INS data fusion algorithm and becomes available for user at maximum rate of 100Hz.

Next output data ratea are available on UART port:: 100, 50, 25, 20, 10, 5 or 2 Hz or on as-requested basis using the GP, 'Get Packet' command. The digital IMU data is available in Scaled Sensor Data ('S1' Packet) or in INS Solution Data ('E3' packet). In the data is output in scaled engineering units. See section 5 of the manual for full packets descriptions.

3.5. User Settings

Selecting which sensors are included in the output can be accomplished using either the write-field or set-field command (WF or SF) over UART or SPI interface. For UART using the SF command will cause the setting to take effect immediately, but this value is not stored in non-volatile memory; the WF command will require a unit-reset before the output is modified but is stored in non-volatile memory. Several example commands follow:

To get the values for fields 0x42 and 0x43 stored in RAM, use the Get-Field (GF) command:

5555 4746 05 02 0042 0043 a0d0

To set the value for field 0x43 stored in RAM to 0x1 (output sensor #1 only), use the Set-Field (SF) command:

5555 5346 05 01 0043 0001 236d

To change the value for field 0x42 in EEPROM to 0x1 (enable sensor #1 only), use the Write-Field (WF) command:

5555 5746 05 01 0042 0001 1b30

4. UART interface definition

This section of the manual explains the INS330BI packet formats as well as the supported commands. NAV-VIEW also features a number of tools that can help a user understand the packet types available and the information contained within them. This section of the manual assumes that the user is familiar with ANSI C programming language and data type conventions.

For information required to parse input data packets, please see refer to Appendix C: Sample Packet Decoding.

4.1. General settings

Standard serial port settings are used with 1 start bit, 8 data bits, no parity bit, 1 stop bit, and no flow control. Standard baud rates supported are: 38400, 57600, 115200, 230400. The default baud rate for unit in Slave mode is 230400. In Standalone and Master modes, default baudrate is 115200.

Common definitions include:

- A word is defined to be 2 bytes or 16 bits.
- All communications to and from the unit are packets that start with a single word alternating bit preamble, 0x5555, which corresponds to a sequence of two ASCII 'U 'characters.
- All multiple byte values are transmitted in a Big-Endian format (Most Significant Byte First).
- All communication packets end with a single word CRC (2 bytes). CRC's are calculated on all packet bytes excluding the preamble and CRC itself. Input packets with incorrect CRC's will be ignored.
- Each complete communication packet must be transmitted to the INS330BI Series inertial system within a 4 second period.

4.2. Number formats

Number Format Conventions include:

- 0x as a prefix to hexadecimal values
- single quotes ('') to delimit ASCII characters
- no prefix or delimiters to specify decimal values.

The following table defines number formats:

Table 2: Number formats

Descriptor	Description	Size (bytes)	Comment	Range
U1	Unsigned Char	1		0 to 255
U2	Unsigned Short	2		0 to 65535
U4	Unsigned Integer	4		0 to 2^32-1
I2	Signed Short	2	2's Complement	-2^15 to 2^15-1
I2*	Signed Short	2	Shifted 2's Complement	Shifted to specified range
I4	Signed Integer	4	2's Complement	-2^31 to 2^31-1
F4	Floating Point	4	IEEE754 Single Precision	-1*2^127 to 2^127

Descriptor	Description	Size (bytes)	Comment	Range
SN	String	N	ASCII	

4.3. Packet format

All Input and Output packets conform to the following structure:

0x5555	*<2-byte packet type (U2)>*	*<payload byte-length (U1)>*	*<variable length payload>*	*<2-byte CRC (U2)>*

4.3.1. *Packet header*

The packet header is always the bit pattern 0x5555.

4.3.2. *Packet type*

The packet type is always two bytes long in unsigned short integer format. Most input and output packet types can be interpreted as a pair of ASCII characters. As a semantic aid consider the following single character acronyms:

P = packet

F = fields

Refers to Fields which are settings or data contained in the unit

E = EEPROM

Refers to factory data stored in EEPROM

R = read

Reads default non-volatile fields

G = get

Gets current volatile fields or settings

W = write

Writes default non-volatile fields. These fields are stored in non-volatile memory and determine the unit's behavior on power up. Modifying default fields take effect on the next power up and thereafter.

S = set

Sets current volatile fields or settings. Modifying current fields will take effect immediately by modifying internal RAM and are lost on a power cycle.

4.3.3. *Payload length*

The payload length is always a one-byte unsigned character with a range of 0-255. The payload length byte is the length (in bytes) of the *<variable length payload>* portion of the packet only and does not include the CRC.

4.3.4. *Payload*

The payload is of variable length based on the packet type.

4.3.5. 16-bit CRC-CCITT

Packets end with a 16-bit CRC-CCITT calculated on the entire packet excluding the 0x5555 header and the CRC field itself. A discussion of the 16-bit CRC-CCITT and sample code for implementing the computation of the CRC is included at the end of this document. This 16-bit CRC standard is maintained by the International Telecommunication Union (ITU). The highlights are:

Width = 16 bits

Polynomial 0x1021

Initial value = 0xFFFF

No XOR performed on the final value.

See Appendix B for sample code that implements the 16-bit CRC algorithm.

4.3.6. Messaging overview

Table 3 below summarizes the INS330BI UART messages. Packet types are assigned mostly using the ASCII mnemonics defined above and are indicated in the summary table below and in the detailed sections for each command. The payload byte-length is often related to other data elements in the packet as defined in the table below. The referenced variables are defined in the detailed sections following. Output messages are sent from the INS330BI Series inertial system to the user system as a result of a poll request or a continuous packet output setting. Input messages are sent from the user system to the INS330BI Series inertial system and will result in an associated Reply Message or NAK message. Note that reply messages typically have the same *<2-byte packet type (U2)>* as the input message that evoked it but with a different payload.

Table 3: Message table

ASCII Mnemonic	***<2-byte packet type (U2)>***	***<payload byte-length (U1)>***	**Description**	**Type**
Link Test				
PK	0x504B	0	Ping Command and Response	Input/Reply Message
Interactive Commands				
GP	0x4750	2	Get Packet Request	Input Message
NAK	0x1515	2	Error Response	Reply Message
Output Messages: Status & Other, (Polled Only)				
ID	0x4944	5+N	Identification Data	Output Message
VR	0x5652	5	Version Data	Output Message

ASCII Mnemonic	*<2-byte packet type (U2)>*	*<payload byte-length (U1)>*	**Description**	**Type**
T0	0x5430	28	Test 0 (Detailed BIT and Status)	Output Message
Output Messages: Measurement Data (Continuous or Polled)				
S0	0x5330	30	Scaled Sensor 0 Data	Output Message
S1	0x5331	24	Scaled Sensor 1 Data	Output Message
E3	0x4533	32	INS solution Data	Output Message
MG	0x4D47	28	Master Data	Output Message from master to slave unit
SA	0x4D47	18	Slave Data	Output Message from slave to master unit
Advanced Commands				
WF	0x5746	numFields*4+1	Write Fields Request	Input Message
WF	0x5746	numFields*2+1	Write Fields Response	Reply Message
SF	0x5346	numFields*4+1	Set Fields Request	Input Message
SF	0x5346	numFields*2+1	Set Fields Response	Reply Message
RF	0x5246	numFields*2+1	Read Fields Request	Input Message
RF	0x5246	numFields*4+1	Read Fields Response	Reply Message
GF	0x4746	numFields*2+1	Get Fields Request	Input Message
GF	0x4746	numFields*4+1	Get Fields Response	Reply Message

5. Standard UART commands and messages

This section describes the functionality and operation of the UART port.

5.1. Link test

5.1.1. *Ping command*

Ping ('PK' = 0x504B)			
Preamble	**Packet Type**	**Length**	**Termination**
0x5555	0x504B	0x00	<CRC (U2)>

The ping command has no payload. Sending the ping command will cause the unit to send a ping response.

Ping response

Ping ('PK' = 0x504B)			
Preamble	**Packet Type**	**Length**	**Termination**
0x5555	0x504B	0x00	<CRC (U2)>

The unit will send this packet in response to a ping command.

5.2. Interactive commands

Interactive commands are used to interactively request data from the INS330BI, and to calibrate or reset the INS330BI.

5.2.1. *Set Master Unit Mode "MM" Packet*

Get Packet ('MM' = 0x4D4D)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4D4D	0x01	<MM payload>	<CRC (U2)>

The payload is single byte command which may have next values:

0x01 – Enter Transparent mode. Upon reception of this command master unit enters transparent mode and slave unit can be reached via master unit. Can be configured or FW can be updated.

0x02 – Enter normal mode. . Upon reception of this command master unit returns back from transparent to normal mode

5.2.2. *Get Packet request*

Get Packet ('GP' = 0x4750)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4750	0x02	<GP payload>	<CRC (U2)>

This command allows the user to poll for both measurement packets and special purpose output packets including 'T0', 'VR', and 'ID'.

GP payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	requestedPacketType	U2	-	-	The requested packet type

Refer to the sections below for Packet Definitions sent in response to the 'GP' command.

5.2.3. *Error response*

Error response (ASCII NAK, NAK = 0x1515)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x1515	0x02	<NAK payload>	<CRC (U2)>

The unit will send this packet in place of a normal response to a *failedInputPacketType* request if it could not be completed successfully.

NAK payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	failedInputPacketType	U2	-	-	the failed request

5.3. Output packets (Polled)

The following packet formats are special informational packets which can be requested using the 'GP ' command.

5.3.1. *Identification data packet*

Identification Data ('ID' = 0x4944)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4944	5+N	<ID payload>	<CRC (U2)>

This packet contains the unit *serialNumber* and *modelString*. The model string is terminated with 0x00. The model string contains the programmed versionString (8-bit ASCII values) followed by the firmware part number string delimited by a whitespace.

ID payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	serialNumber	U4	-	-	Unit serial number
4	modelString	SN	-	-	Unit Version String
4+N	0x00	U1	-	-	Zero Delimiter

5.3.2. *Version data packet*

Version Data ('VR' = 0x5652)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5652	5	<VR payload>	<CRC (U2)>

This packet contains firmware version information. *majorVersion* changes may introduce serious incompatibilities. *minorVersion* changes may add or modify functionality, but maintain backward compatibility with previous minor versions. *patch* level changes reflect bug fixes and internal modifications with little effect on the user. The build *stage* is one of the following: 0=release candidate, 1=development, 2=alpha, 3=beta. The *buildNumber* is incremented with each engineering firmware build. The *buildNumber* and *stage* for released firmware are both zero. The final beta candidate is v.w.x.3.y, which is then changed to v.w.x.0.1 to create the first release candidate. The last release candidate is v.w.x.0.z, which is then changed to v.w.x.0.0 for release.

VR payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	majorVersion	U1	-	-	Major firmware version
1	minorVersion	U1	-	-	Minor firmware version
2	patch	U1	-	-	Patch level
3	stage	-	-	-	Development Stage (0=release candidate, 1=development, 2=alpha, 3=beta)
4	buildNumber	U1	-	-	Build number

5.3.3. *Test 0 (Detailed BIT and status) packet*

Test ('T0' = 0x5430)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
03.3x5555	0x5430	0x1C	<T0 payload>	<CRC (U2)>

This packet contains detailed BIT and status information. The full BIT Status details are described in Section 7 of this manual.

T0 payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	BITstatus	U2	-	-	Master BIT and Status Field
2	hardwareBIT	U2	-	-	Hardware BIT Field

4	Reserved	U2	-	-	
6	Reserved	U2	-	-	
8	Reserved	U2	-	-	
10	Reserved	U2	-	-	
12	Reserved	U2	-	-	
14	softwareBIT	U2	-	-	Software BIT Field
16	softwareAlgorithmBIT	U2	-	-	Software Algorithm BIT Field
18	softwareDataBIT	U2	-	-	Software Data BIT Field
20	hardwareStatus	U2	-	-	Hardware Status Field
22	comStatus	U2	-	-	Communication Status Field
24	softwareStatus	U2	-	-	Software Status Field
26	sensorStatus	U2	-	-	Sensor Status Field

5.4. Output packets (polled or continuous)

NOTE
Please see scaling information for each sensor. Specifically, the sensor data in the SPI messages and the UART messages are scaled differently.

5.4.1. INS Solution Packet E3 (default output packet to user)

Default Data ('E3' = 0x4533)				
Preamble	Packet Type	Length	Payload	Termination
0x5555	0x4533	0x20	<data payload >	<CRC (U2)>

This packet contains scaled sensor data. The scaled sensor data is fixed point, 2 bytes per sensor, MSB first, for the sensors in the following order: accels(x,y,z); gyros(x,y,z); temps(x,y,z,board). Data involving angular measurements include the factor pi in the scaling and can be interpreted in either radians or degrees. Note the timer value can be used for synchronization and computation of DeltaT. It may appear in NAV-VIEW log files under another column heading.

Angular rates: scaled to a range of [-630 deg/sec, +630 deg/sec]

Accelerometers: scaled to a range of [-10,+10] g

Temperature: scaled to a range of [-100, +100] °C

Payload definition

Byte Offset	Name	Format	Notes	Scaling	unit	Description
0	Counter	U4	MSB first	1	ms	Unsigned int, 4 bytes
4	Roll	I2	MSB first	360° /2^16	°	
6	Pitch	I2	MSB first	360° /2^16	°	
8	Yaw	I2	MSB first	360° /2^16	°	
10	Steering angle of front wheel	I2	MSB first	360° /2^16	°	
12	Accel_X_Master	I2	MSB first	20/2^16	g	
14	Accel_Y_Master	I2	MSB first	20/2^16	g	
16	Accel_Z_Master	I2	MSB first	20/2^16	g	
18	Gyro_X_Master	I2	MSB first	1260°/2^16	°/sec	
20	Gyro_Y_Master	I2	MSB first	1260°/2^16	°/sec	
22	Gyro_Z_Master	I2	MSB first	1260°/2^16	°/sec	
24	Steering angle rate	I2	MSB first	1260°/2^16	°/sec	Steering angle rate of front wheel
26	Vehicle speed	I2	MSB first	0.001	m/s	Positive and negative value to show speed of advancing or retreating will be better
28	Master algorithm states	2 bytes	MSB first			
30	Slave algorithm states	2 bytes	MSB first			

Table 4 Master Algorithm States

Bit address and name	Value	description
0 Algorithm uninitialized	0	The algorithm is initialized.
	1	The algorithm is not initialized.
1 High gain mode	0	The algorithm is not in high gain mode.
	1	The algorithm is in high gain mode.
2 Attitude only	0	The algorithm is in INS mode.
	1	The algorithm is in attitude only mode.

3 Turn switch	0	The unit is not taking a turn.
	1	The unit is taking a turn.
4 Linear accel	0	Linear accel is not detected.
	1	Linear accel is detected.
5 Vehilce is motionless #1	0	The vehicle is detected not to be motionless by IMU data.
	1	The vehicle is detected to be motionless by IMU data
6 Vehilce is motionless #2	0	The vehicle is detected not to be motionless by GNSS data.
	1	The vehicle is detected to be motionless by GNSS data
7 GNSS heading valid	0	GNSS heading is not valid.
	1	GNSS heading is valid.
8 Heading lock	0	Heading is not locked.
	1	Heading is locked to its current value because the vehicle is motionless.
9 PPS available	0	PPS signal is not available.
	1	PPS is available.
10 Algorithm synced to PPS	0	The algorithm is not synced to PPS.
	1	The algorithm is synced to PPS.
11:12 Heading initialization status	0	Heading is not initialized.
	1	Heading is initialized using magnetometer. (not used in INS330BI)
	2	Heading is quickly (maybe inaccurately) initialized using GNSS.
	3	Heading is reliably initialized using GNSS.
13:15 Reserved		

Table 5 Steering **Angle** Algorithm States

Bit address and name	Value	description
0:2 Algorithm state	0	Waiting for the sensor data to get stabilized.
	1	The algorithm is calculating initial gyro bias.
	2	The algorithm starts to calculate the steering angle.
3:5 Steering angle initialization	0	The steering angle is not initialized.
	1	The steering angle is quickly initialized with a less accurate value.
	2	The steering angle is quickly initialized with an accurate value.
6 Vehilce is motionless #1	0	The vehicle is detected not to be motionless by IMU data.
	1	The vehicle is detected to be motionless by IMU data
7 Turn switch	0	The unit is not taking a turn.
	1	The unit is taking a turn.
8 Vehilce is motionless #2	0	The vehicle is detected not to be motionless by GNSS data.
	1	The vehicle is detected to be motionless by GNSS data
9 GNSS outage	0	There is not a long-time GNSS outage. (long time: default 20sec)
	1	There is a long-time GNSS outage.
10 Steering angle locked	0	The steering angle is not locked to its current value.
	1	The steering angle is locked to its current value.

11 Motion anomaly	0	There is no anomaly in vehicle motion.
	1	There is anomaly in vehicle motion.
12 GNSS heading valid	0	There is no valid GNSS data since power on.
	1	There is at least one set of valid GNSS data since power on.
13 Initial gyro bias valid	0	Initial gyro bias is not valid.
	1	Initial gyro bias is valid.
14:15 Reserved		

5.4.2. *Master Data Packet MG (output from master unit to slave unit)*

Default Data ('MG' = 0x4D47)				
Preamble	Packet Type	Length	Payload	Termination
0x5555	0x4D47	0x1C	<data payload >	<CRC (U2)>

Byte Offset	Name	Format	Notes	Scaling	unit	Description
0	Counter	U4	MSB first	1	ms	Unsigned int, 4 bytes
4	Accel_X_Master	I2	MSB first	20/2^16	g	
6	Accel_Y_Master	I2	MSB first	20/2^16	g	
8	Accel_Z_Master	I2	MSB first	20/2^16	g	
10	Gyro_X_Master	I2	MSB first	1260°/2^16	°/sec	
12	Gyro_Y_Master	I2	MSB first	1260°/2^16	°/sec	
14	Gyro_Z_Master	I2	MSB first	1260°/2^16	°/sec	
16	GNSS time of week	U4	MSB first		ms	This value can be acquired from the master GNSS driver.
20	Ground speed	I2	MSB first	0.001	m/s	Signed. The customer defines a speed limit of [-100, 100]km/h, which can be covered by the 16bit signed integer with a scaling of 0.001m/s. The speed accuracy is 0.03~0.05m/s, which can also be well handled by the scaling. This value can be acquired from the master GNSS driver.
22	GNSS update flag	Char				No-zero value indicates the GNSS info is updated in this packet.
23	GNSS fix type	Char				Zero indicates invalid GNSS info. This value can be acquired from the master GNSS driver.

24	Reserved	4 bytes	MSB first			Reserved for future use.

5.4.3. *Slave Data Packet "SA" (output from slave to master unit)*

Default Data ('SA' = 0x5341)				
Preamble	Packet Type	Length	Payload	Termination
0x5555	0x5341	0x12	<data payload >	<CRC (U2)>

Byte Offset	Name	Format	Notes	Scaling	unit	Description
0	Counter	U4	MSB first	1	ms	Unsigned int, 4 bytes
4	Steering angle	I2	MSB first	360° /2^16	°	Steering angle of front wheel
6	Steering angle rate	I2	MSB first	1260°/2^16	°/sec	Steering angle rate of front wheel
8	Algorithm states	U2	MSB first			Refer to 错误!未找到引用源。 for details
10	Reserved	8 bytes	MSB first			Reserved for future use.

5.4.4. *Scaled sensor data packet 1 (IMU data packet)*

◀ NOTE
Sensor data scaling is specific to the message or register being read. Please see scaling information for each sensor data read operation. Specifically, the sensor data in the SPI S0_BURST, the SPI S1_BURST, the UART S0 message and the UART S1 message are scaled differently.

Scaled sensor data ('S1' = 0x5331)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5331	0x18	<S1 payload>	<CRC (U2)>

This packet contains scaled sensor data. Data involving angular measurements include the factor pi in the scaling and can be interpreted in either radians or degrees. Note the timer value can be used for synchronization and computation of Delta T. It may appear in NAV-VIEW log files under another column heading.

Angular rates: scaled to range of -630 deg/sec to +630 deg/sec)

Accelerometers: scaled to a range of [-10,+10)g

Temperature: scaled to a range of [-100, +100]°C

S1 payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	xAccel	I2	20/2^16	g	X accelerometer
2	yAccel	I2	20/2^16	g	Y accelerometer
4	zAccel	I2	20/2^16	g	Z accelerometer
6	xRate	I2	1260°/2^16	°/sec	X angular rate
8	yRate	I2	1260°/2^16	°/sec	Y angular rate
10	zRate	I2	1260°/2^16	°/sec	Z angular rate
12	xRateTemp	I2	200/2^16	deg. C	X rate temperature
14	yRateTemp	I2	200/2^16	deg. C	Y rate temperature
16	zRateTemp	I2	200/2^16	deg. C	Z rate temperature
18	boardTemp	I2	200/2^16	deg. C	CPU board temperature
20	timer	U2	15.259022	uS	Free running fast counter 1s= 65535, captured at sampling
22	BITstatus	U2	-	-	Master BIT and Status

6. Advanced UART commands

The advanced commands allow users to change the INS330BI Series settings.

Section 6.1 below describes all available configuration settings available through the UART interface and section 6.2 below describes different ways to send commands to read/write configuration settings.

6.1. Configuration fields

Configuration fields determine various behaviors of the unit that can be modified by the user. These include settings like baud rate, packet output rate and type, algorithm type, etc. These fields are stored in the EEPROM and loaded on power up. They can be read from the EEPROM using the 'RF' command. These fields can be written to the EEPROM affecting the default power up behavior using the 'WF' command. The current values of these fields (which may be different from the values stored in the EEPROM) can also be accessed using the 'GF' command. All of these fields can also be modified immediately for the duration of the current power cycle using the 'SF' command. The unit will always power up in the configuration stored in the EEPROM. Configuration fields can only be set or written with valid data from the table below.

Table 6: Configuration fields

Configuration fields	Field ID	Valid Values	Description
Packet rate divider	0x0001	0x0, 0x01(default), 0x02, 0x04, 0x05, 0x0A, 0x14, 0x19, 0x32	quiet, 100Hz (default), 50Hz, 25Hz, 20Hz, 10Hz, 5Hz, 4Hz, 2Hz
User UART BAUD rate (applicable only to standalone unit)	0x0002	0x02, 0x03, 0x05, 0x06(default), 0x07, 0x08	38400, 57600, 115200 (default for master unit) , 230400 (default for slave unit), baud rate
Cont packet type	0x0003	0x5330, 0x5331, 0x4533(default)	Packet Types: S0, S1, E3(default)
Reserved	0x0004	NA	NA
Accelerometer filter setting	0x0005	0x00 (0): Unfiltered 0x68A1 (26785): 2 Hz 0x29D9 (10713): 5 Hz 0x14EC (5356): 10 Hz 0xA76 (2678): 20 Hz 0x85E (2142): 25 Hz 0x53A (1338): 40 Hz 0x42E (1070): 50 Hz	Sets low pass cutoff for accelerometers. Default – 25 Hz
Gyro filter setting	0x0006	0x00 (0): Unfiltered 0x68A1 (26785): 2 Hz 0x29D9 (10713): 5 Hz 0x14EC (5356): 10 Hz 0xA76 (2678): 20 Hz 0x85E (2142): 25 Hz 0x53A (1338): 40 Hz 0x42E (1070): 50 Hz	Sets low pass cutoff for rate sensors. Default 25 Hz

Orientation	0x0007	See Table 9 below	Determine forward, rightward, and downward facing sides Default is 0x0000
Lever arm X	0x0056	0	Dimensions from the Master device to the to the Antenna, axis X (in mm)
Lever arm Y	0x0057	0	Dimensions from the Master device to the to the Antenna, axis Y (in mm)
Lever arm Z	0x0058	0	Dimensions from the Master device to the to the Antenna, axis Z (in mm)
Slave lever arm X	0x005C	0	Dimensions from the center of the two rear wheels to the GNSS antenna, axis X (in mm)
Slave lever arm Y	0x005D	0	Dimensions from the center of the two rear wheels to the GNSS antenna, axis Y (in mm)
Wheel Distance	0x005B	0	Distance between the front wheel and the rear wheel (in mm)
GNSS ODR	0x005A	10	GNSS module ODR in units of Hz.
Internal packets (applicable only to Master mode unit)	0x0059	0x0000 0x4D47 MG packet	Config user port of master unit, and send MG packets in user port.

NOTE
User UART BAUD rate can be changed by command 'WF' and will take effect after unit reset or power cycle.

6.1.1. *Packet rate divider*

This field configures the output packet rate of the UART interface. Packet rates supported by the unit and their associated Packet rate divider field values are mentioned in Table 6 above.

6.1.2. *User UART baudrate*

The unit baudrate field can only be set permanently and is effective only after power cycle or reset of the unit. Baud-rates supported by the unit and associated values are mentioned in Table 6 above.

6.1.3. *Continuous packet type field*

When the Packet rate divider field is not set to *quiet*, the unit continuously sends data packets at the rate set in the Packet rate divider field. This field selects the packet type the unit sends. Few packet types are supported by INS330BI unit. Packet type S0 is selected by writing value 0x5330, packet type S1 is selected by writing 0x5331 and packet type E3 is selected by writing 0x4543 to the Continuous packet type field.

The supported packet depends on the model number. Please refer to Section 5.3 for a complete list of the available packet types.

6.1.4. *Digital filter settings*

These two fields set the digital low pass filter cutoff frequencies (see below table)

Table 7: Digital filter settings

Function	Field number
FilterAccel	0x05 See Table 6 for values
FilterGyro	0x06 See Table 6 for values

6.1.5. *Orientation field*

This field controls the orientation of the user-specified axes (+X, +Y, +Z) relative to the nominal factory-axes (+Ux, +Uy, +Uz). The default coordinate frame is (+X, +Y, +Z) = (Ux, Uy, Uz), corresponding to an Orientation Field setting of 0x0000.

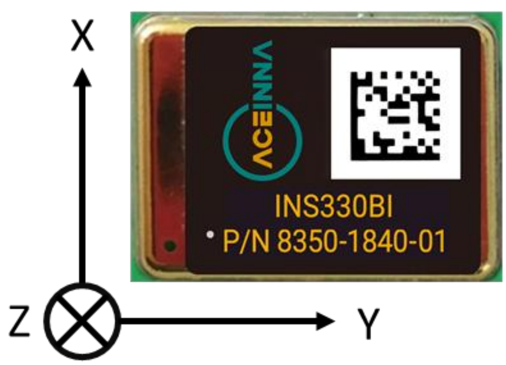


Figure 9: INS330BI Default Orientation Field (0x0000)

To set the coordinate-frame according to your needs, specify the direction of the +X, +Y, +Z axis relative to +Ux, +Uy, +Uz (see Table 8). The complete list of accepted orientations (right-handed coordinate frames) are listed in Table 9.

Table 8: Orientation definitions

Description	Bits	Meaning
X Axis Sign	0	0 = positive, 1 = negative
X Axis	1:2	0 = Ux, 1 = Uy, 2 = Uz, 3 = N/A
Y Axis Sign	3	0 = positive, 1 = negative
Y Axis	4:5	0 = Uy, 1 = Uz, 2 = Ux, 3 = N/A
Z Axis Sign	6	0 = positive, 1 = negative
Z Axis	7:8	0 = Uz, 1 = Ux, 2 = Uy, 3 = N/A
Reserved	9:15	N/A

There are 24 possible orientation configurations listed in Table 9. Setting/Writing the field to anything else generates a NAK and has no effect.

Table 9: Orientation fields

Orientation field value	X Axis	Y Axis	Z Axis
0x0000 (default orientation)	+Ux	+Uy	+Uz
0x0009	-Ux	-Uy	+Uz
0x0023	-Uy	+Ux	+Uz
0x002A	+Uy	-Ux	+Uz
0x0041	-Ux	+Uy	-Uz
0x0048	+Ux	-Uy	-Uz
0x0062	+Uy	+Ux	-Uz
0x006B	-Uy	-Ux	-Uz
0x0085	-Uz	+Uy	+Ux
0x008C	+Uz	-Uy	+Ux
0x0092	+Uy	+Uz	+Ux
0x009B	-Uy	-Uz	+Ux
0x00C4	+Uz	+Uy	-Ux
0x00CD	-Uz	-Uy	-Ux
0x00D3	-Uy	+Uz	-Ux
0x00DA	+Uy	-Uz	-Ux
0x0111	-Ux	+Uz	+Uy
0x0118	+Ux	-Uz	+Uy
0x0124	+Uz	+Ux	+Uy
0x012D	-Uz	-Ux	+Uy
0x0150	+Ux	+Uz	-Uy
0x0159	-Ux	-Uz	-Uy
0x0165	-Uz	+Ux	-Uy
0x016C	+Uz	-Ux	-Uy

6.2. Read/Write of configuration fields

Each configuration field mentioned in the section above can be set permanently or temporarily using *WF* and *SF* commands respectively. Permanent and temporary configurations can be read using *RF* and *GF* commands respectively.

More information on each of these four commands is given in below.

6.2.1. *Write Fields command*

Write Fields ('WF' = 0x5746)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5746	1+numFields*4	<WF payload>	<CRC (U2)>

This command allows the user to change the default power-up configuration fields. Writing the default configuration will not take effect until the unit is power cycled. *NumFields* is the number of words to be written. The *field0, field1, etc.* are the field IDs that will be written with the *field0Data, field1Data, etc.*, respectively. The unit will not write to calibration or algorithm fields. If at least one field is successfully written, the unit will respond with a write fields response containing the field IDs of the successfully written fields. If any field is unable to be written, the unit will respond with an error response.

◀ NOTE
Both a write fields and an error response may be received as a result of a write fields command. Attempting to write a field with an invalid value is one way to generate an error response. A table of field IDs and valid field values is available in section 6.1.

WF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	numFields	U1	-	-	The number of fields to write
1	field0	U2	-	-	The first field ID to write
3	field0Data	U2	-	-	The first field ID's data to write
5	field1	U2	-	-	The second field ID to write
7	field1Data	U2	-	-	The second field ID's data
…	…	U2	-	-	…
numFields*4 -3	field…	U2	-	-	The last field ID to write
numFields*4 -1	field…Data	U2	-	-	The last field ID's data to write

6.2.2. *Write Fields response*

Write Fields ('WF' = 0x5746)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5746	1+numFields*2	<WF payload>	<CRC (U2)>

The unit will send this packet in response to a write fields command if the command has completed without errors.

WF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	numFields	U1	-	-	The number of fields written
1	field0	U2	-	-	The first field ID written
3	field1	U2	-	-	The second field ID written
…	…	U2	-	-	More field IDs written
numFields*2 – 1	Field…	U2	-	-	The last field ID written

6.2.3. *Set Fields command*

Set Fields ('SF' = 0x5346)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5346	1+numFields*4	<SF payload>	<CRC (U2)>

This command allows the user to change a unit's current configuration (SF) fields immediately. The settings will be lost after the unit is power cycled or reset. *NumFields* is the number of words to be set. The *field0, field1, etc.* are the field IDs that will be written with the *field0Data, field1Data, etc.*, respectively. The command will not affect calibration or algorithm fields. If at least one field is successfully set, the unit will respond with a set fields response containing the field IDs of the successfully set fields. If any field is unable to be set, the unit will respond with an error response.

NOTE
Set fields response and or error response may be received as a result of set fields command. Attempts to set a field with an invalid value is one way to generate an error response. A table of field IDs and valid field values is available in Section 6.1.

SF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	numFields	U1	-	-	The number of fields to set
1	field0	U2	-	-	The first field ID to set
3	field0Data	U2	-	-	The first field ID's data to set
5	field1	U2	-	-	The second field ID to set
7	field1Data	U2	-	-	The second field ID's data to set
…	…	U2	-	-	…
numFields*4 -3	field…	U2	-	-	The last field ID to set

numFields*4 -1	field…Data	U2	-	-	The last field ID's data to set

6.2.4. *Set Fields response*

Set Fields ('SF' = 0x5346)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5346	1+numFields*2	<SF payload>	<CRC (U2)>

The unit will send this packet in response to a set fields command if the command has completed without errors.

SF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	numFields	U1	-	-	The number of fields set
1	field0	U2	-	-	The first field ID set
3	field1	U2	-	-	The second field ID set
…	…	U2	-	-	More field IDs set
numFields*2 - 1	Field…	U2	-	-	The last field ID set

6.2.5. *Read Fields command*

Read Fields ('RF' = 0x5246)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5246	1+numFields*2	<RF payload>	<CRC (U2)>

This command allows the user to read the default power-up configuration fields. *NumFields* is the number of fields to read. The *field0, field1, etc.* are the field IDs to read. RF may be used to read configuration and calibration fields. If at least one field is successfully read, the unit will respond with a read fields response containing the field IDs and data from the successfully read fields. If any field is unable to be read, the unit will respond with an error response.

◀ NOTE
Both a read field and an error response may be received as a result of a read fields command.

RF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	numFields	U1	-	-	The number of fields to read
1	field0	U2	-	-	The first field ID to read
3	field1	U2	-	-	The second field ID to read

…	…	U2	-	-	More field IDs to read
numFields*2 - 1	Field…	U2	-	-	The last field ID to read

6.2.6. *Read Fields response*

Read Fields ('RF' = 0x5246)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x5246	1+numFields*4	<RF payload>	<CRC (U2)>

The unit will send this packet in response to a read field request if the command has completed without errors.

RF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	numFields	U1	-	-	The number of fields read
1	field0	U2	-	-	The first field ID read
3	field0Data	U2	-	-	The first field ID's data read
5	field1	U2	-	-	The second field ID read
7	field1Data	U2	-	-	The second field ID's data read
…	…	U2	-	-	…
numFields*4 -3	field…	U2	-	-	The last field ID read
numFields*4 -1	field…Data	U2	-	-	The last field ID's data read

6.2.7. *Get Fields command*

Get Fields ('GF' = 0x4746)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4746	1+numFields*2	<GF Data>	<CRC (U2)>

This command allows the user to get the unit's current parameters. *NumFields* is the number of fields to get. The *field0, field1, etc.* are the field IDs to get. GF may be used to get configuration, calibration, and algorithm fields from RAM. Multiple algorithm fields will not necessarily be from the same algorithm iteration. If at least one field is successfully collected, the unit will respond with a get fields response with data containing the field IDs of the successfully received fields. If any field is unable to be received, the unit will respond with an error response. Note

NOTE
Both a get fields and an error response may be received as the result of a get fields command.

GF payload contents

Byte Offset	Name	Format	Scaling	Units	Description
0	numFields	U1	-	-	The number of fields to get
1	field0	U2	-	-	The first field ID to get
3	field1	U2	-	-	The second field ID to get
…	…	U2	-	-	More field IDs to get
numFields*2 – 1	Field…	U2	-	-	The last field ID to get

6.2.8. *Get Fields response*

Get Fields (‘GF’ = 0x4746)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4746	1+numFields*4	<GF Data>	<CRC (U2)>

The unit will send this packet in response to a get fields request if the command has completed without errors.

GF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	numFields	U1	-	-	The number of fields retrieved
1	field0	U2	-	-	The first field ID retrieved
3	field0Data	U2	-	-	The first field ID’s data retrieved
5	field1	U2	-	-	The second field ID retrieved
7	field1Data	U2	-	-	The second field ID’s data
…	…	U2	-	-	…
numFields*4 -3	field…	U2	-	-	The last field ID retrieved
numFields*4 -1	field…Data	U2	-	-	The last field ID’s data retrieved

6.2.9. *Entry transportation mode(only applicable for master unit)*

The unit will receive set cmd in current baud rate(such as: 115200), and it will shift normal working mode to transportation mode until cancel cmd is received or power reset.

During transportation mode, it means bridge mode, build communication between external MCU and slave unit.

It will receive the data from external MCU(115200) and transfer same data to slave module(230400), or receive data from slave module and transfer msg to external MCU.

Entry Transportaion Mode				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4D4D	1	< Data>	<CRC (U2)>

The unit will send this packet in response to a get fields request if the command has completed without errors.

GF payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	cmd	-	-	-	01-enter enter transparent mode 02-exit transparent mode

7. Advanced UART Port BIT

7.1. Built In Test (BIT) and Status fields

Internal health and status are monitored and communicated in both hardware and software. The ultimate indication of a fatal problem is a hardware BIT signal on the user connector which is mirrored in the software BIT field as the masterFail flag. This flag is thrown as a result of a number of instantly fatal conditions (known as a "hard" failure) or a persistent serious problem (known as a "soft" failure). Soft errors are those which must be triggered multiple times within a specified time window to be considered fatal. Soft errors are managed using a digital high-pass error counter with a trigger threshold.

The masterStatus flag is a configurable indication as determined by the user. This flag is asserted as a result of any asserted alert signals which the user has enabled.

The hierarchy of BIT and Status *fields* and signals is depicted here:

BIT Status Field

1. masterFail
   - A. HardwareError (hardwareBIT Field)
     - a. HwTestError
     - b. AccelerometerError
     - c. RateSensorError
   - B. SoftwareError (softwareBIT Field)
     - a. calibrationCRCError
2. masterStatus
   - A. HardwareStatus (hardwareStatus Field)
     - a. unlockedEEPROM
     - b. AccelerometerFaultDetected
     - c. RateSensorFaultDetected
   - B. SoftwareStatus (softwareStatus Field)
     - a. Factory Test Mode
   - C. SensorStatus (sensorStatus Field)
     - a. overRange (enabled by default)

7.2. Master BIT and Status (BITstatus) field

The BITstatus field is the global indication of health and status of the INS330BI Series product (See

Table 10). The LSB contains BIT information and the MSB contains status information.

There are four intermediate signals that are used to determine when masterFail and the hardware BIT signal are asserted. These signals are controlled by various systems checks in software that are classified into three categories: hardware, communication, and software. Instantaneous soft failures in each of these four categories will trigger these intermediate signals but will not trigger the masterFail until the persistency conditions are met.

There are four intermediate signals that are used to determine when the masterStatus flag is asserted: hardwareStatus, sensorStatus, comStatus, and softwareStatus. masterStatus is the logical OR of these intermediate signals. Each of these intermediate signals has a separate field with individual indication flags.

Table 10: BIT Status field

BITstatus Field	Bits	Meaning	Category
masterFail	0	0 = normal, 1 = fatal error has occurred	BIT
HardwareError	1	0 = normal, 1= internal hardware error	BIT
Reserved	2		BIT
softwareError	3	0 = normal, 1 = internal software error	BIT
Reserved	4:7	N/A	
masterStatus	8	0 = nominal, 1 = hardware, sensor, com, or software alert	Status
hardwareStatus	9	0 = nominal, 1 = programmable alert	Status
Reserved	10	N/A	Status
softwareStatus	11	0 = nominal, 1 = programmable alert	Status
sensorStatus	12	0 = nominal, 1 = programmable alert	Status
Reserved	13:15	N/A	

7.3. hardwareBIT field

The hardwareBIT field contains flags that indicate various types of internal hardware errors (see Table 11). Each of these types has an associated message with low level error signals. The hardwareError flag in the BITstatus field is the bitwise OR of this hardwareBIT field.

Table 11: Hardware BIT field

hardwareBIT Field	Bits	Meaning	Category
reserved	0:3	N/A	
accelerometerError	4	0 = normal, 1 = error	Hard
gyroError	5	0 = normal, 1 = error	Hard
Factory selfTestError1	6	Factory use only	Hard
Factory selfTestError2	7	Factory use only	Hard
reserved	8:15	N/A	

7.4. softwareDataBIT field

The softwareDataBIT field contains flags that indicate low level software data errors (see Table 12). The dataError flag in the softwareBIT field is the bitwise OR of this softwareDataBIT field.

Table 12: Software Data BIT field

SoftwareDataBIT field	Bits	Meaning	Category
calibrationCRCError	0	0 = normal, 1 = incorrect CRC on calibration EEPROM data or data has been compromised by a WE command.	Hard
Reserved	2:15	N/A	

7.5. hardwareStatus field

The hardwareStatus field contains flags that indicate various internal hardware conditions and alerts that are not errors or problems (see Table 13).

Table 13: Hardware Status BIT field

hardwareStatus field	Bits	Meaning
unlockedEEPROM	3	0=locked, WE disabled, 1=unlocked, WE enabled
accelFaultDetected	4	1 – accelerometer fault detected
gyroFaultDetected	5	1 – gyro fault detected
Reserved	4:15	N/A

7.6. sensorStatus field

The sensorStatus field contains flags that indicate various internal sensor conditions and alerts that are not errors or problems (see Table 14).

Table 14: Sensor status field

sensorStatus Field	Bits	Meaning
overRange	0	0 = not asserted, 1 = asserted
Reserved	1:15	N/A

8. Bootloader

A bootloader function is available to upgrade the IMU383 firmware. The firmware can be updated over UART following section.

8.1. Updating firmware over UART interface

A user can initiate the bootloader at any time by sending 'JI 'command (see below command's format) to application program. This command forces the unit to enter bootloader mode. The Standalone and Master units will communicate at 115.200Kbps baud rate. Slave unit communicates at 230.400Kbps.

NOTE
In case Slave unit connected to Master unit, firmware in Slave unit can be updated via master unit. In this case before updating FW message "MM" with payload byte equal to 1 (enter transparent mode) needs to be sent to master unit.

As an additional device recovery option immediately after powering up, every INS330BI will enter a recovery window of 100ms prior to application start. During this 100mS window, the user can send 'JI ' command at 115.200Kbs to the Bootloader in order to force the unit to remain in Bootloader mode.

Once the device enters Bootloader mode via the 'JI 'command either during recovery window or from normal operation, a user can send a sequence 'WA 'commands to write a complete application image sequentially into the device's FLASH.

After loading the entire firmware image with successive 'WA 'commands, a 'JA 'command is sent to instruct the unit to exit Bootloader mode and begin application execution. At this point the device will return to its original baud rate.

Optionally, the system can be reboot by toggling power or toggling nRst (pull low and release) to restart the system.

Firmware upgrade is performed by a Write APP command through UART port, through Windows GUI, NAV-View, or a python program. See Appendix A.

The following commands allow users to install a pre-built binary into flash memory and force system enters either bootloader or application mode.

8.1.1. Jump to BootLoader command

Jump to BootLoader ('JI' = 0x4A49)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4A49	0x00		CRC (U2)

The command allows system to enter bootloader mode. Once in bootloader mode, only the bootloading functions are supported.

8.1.2. Write APP command

Write APP ("WA" = 0x5741)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**

0x5555	0x5741	length+5		CRC (U2)

The command allows users to write binary sequentially to flash memory in bootloader mode. The total length is the sum of payload's length and 4-byte address followed by 1-byte data length. See the following table of the payload's format.

WA payload contents					
Byte Offset	**Name**	**Format**	**Scaling**	**Units**	**Description**
0	startingAddress	U4	-	bytes	The FLASH word offset to begin writing data
4	byteLength	U1	-	bytes	The word length of the data to write
5	dataByte0	U1	-	-	FLASH data
6	dataByte1	U1	-	-	FLASH data
…	…				
4+byteLength	dataByte	U1	-	-	FLASH data

Payload starts from 4-byte address of flash memory where the binary is located. The fifth byte is the number of bytes of *dataBytes*, but less than 240 bytes. User must truncate the binary to less than 240-byte blocks and fill each of blocks into payload starting from the sixth byte.

8.1.3. *Jump to Application command*

Jump to Application ('JA' = 0x4A41)				
Preamble	**Packet Type**	**Length**	**Payload**	**Termination**
0x5555	0x4A41	0x00		CRC (U2)

The command allows system to enter application mode.

9. Warranty and support information

9.1. Customer service

As an ACEINNA customer you have access to product support services, which include:

- Single-point return service
- Web-based support service
- Same day troubleshooting assistance
- Worldwide ACEINNA representation
- Onsite and factory training available
- Preventative maintenance and repair programs
- Installation assistance available

9.2. Contact directory

United States: Email: techsupport@aceinna.com

Non-U.S.: Refer to website www.aceinna.com

9.3. Return procedure

9.3.1. *Authorization*

Before returning any equipment, please contact ACEINNA to obtain a Returned Material Authorization number (RMA).

Be ready to provide the following information when requesting an RMA:

- Name
- Address
- Telephone, Fax, Email
- Equipment Model Number
- Equipment Serial Number
- Installation Date
- Failure Date
- Fault Description
- Will it connect to NAV-VIEW 3.X?

9.3.2. *Identification and protection*

If the equipment is to be shipped to ACEINNA for service or repair, please attach a tag TO THE EQUIPMENT, as well as the shipping container(s), identifying the owner. Also indicate the service or repair required, the problems encountered, and other information considered valuable to the service facility such as the list of information provided to request the RMA number.

Place the equipment in the original shipping container(s), making sure there is adequate packing around all sides of the equipment. If the original shipping containers were discarded, use heavy boxes with adequate padding and protection.

9.3.3. Sealing the container

Seal the shipping container(s) with heavy tape or metal bands strong enough to handle the weight of the equipment and the container.

9.3.4. Marking

Please write the words, "FRAGILE, DELICATE INSTRUMENT" in several places on the outside of the shipping container(s). In all correspondence, please refer to the equipment by the model number, the serial number, and the RMA number.

9.3.5. Warranty

The ACEINNA product warranty is one year from date of shipment.

Appendix A: Installation and operation of NAV-VIEW

NAV-VIEW has been designed to allow users to control all aspects of the compatible Aceinna IMU product families. Note some of the functions described are not available on all of the product families (for example, the AHRS or INS functionality does not apply to an IMU). NAV-VIEW version 3.6.2 or higher is required.

NAV-VIEW Computer requirements

The following are minimum requirements for the installation of the NAV-VIEW Software:

- CPU: Pentium-class (1.5GHz minimum)
- RAM Memory: 500MB minimum, 1GB+ recommended
- Hard Drive Free Memory: 20MB
- Operating System: Windows 7 and 10
- Properly installed Microsoft .NET 2.0 or higher

Install NAV-VIEW

To install NAV-VIEW onto your computer:

1. Go to https://www.aceinna.com/manuals and locate the link to the Nav-View zip file. Click the link and download the latest version of the software.
2. Locate the downloaded version of Nav-View on your computer and unzip the file. In the extracted folder click the "setup.exe" file.
3. Follow the setup wizard instructions. You will install NAV-VIEW and .NET 2.0.

⚠ WARNING

Do not reverse the power leads! Reversing the power leads to the INS330BI can damage the unit. Although reverse power protection is provided, ACEINNA is not responsible for resulting damage to the unit should the reverse voltage protection electronics fail.

Setting up NAV-VIEW

With the INS330BI Series product powered up and connected to your PC serial port, open the NAV-VIEW software application.

1. NAV-VIEW should automatically detect the INS330BI Series product and display the serial number and firmware version if it is connected.
2. If NAV-VIEW does not connect, check that you have the correct COM port selected. You will find this under the "Setup" menu. Select the appropriate COM port and allow the unit to automatically match the baud rate by leaving the "Auto: match baud rate" selection marked.
3. If the status indicator at the bottom is green and states, Unit Connected, you're ready to go. If the status indicator doesn't say connected and is red, check the connections between the INS330BI Series product and the computer, check the power supply, and verify that the COM port is not occupied by another device.
4. Under the "View" menu you have several choices of data presentation. Graph display is the default setting and will provide a real time graph of all the INS330BI Series data. The remaining choices will be discussed in the following pages.

Data recording

NAV-VIEW allows the user to log data to a text file (.txt) using the simple interface at the top of the screen. Customers can now tailor the type of data, rate of logging and can even establish predetermined recording lengths.

To begin logging data follow the steps below (See Figure 10):

1. Locate the icon at the top of the page or select "Log to File" from the "File" drop down menu.
2. The following menu will appear.

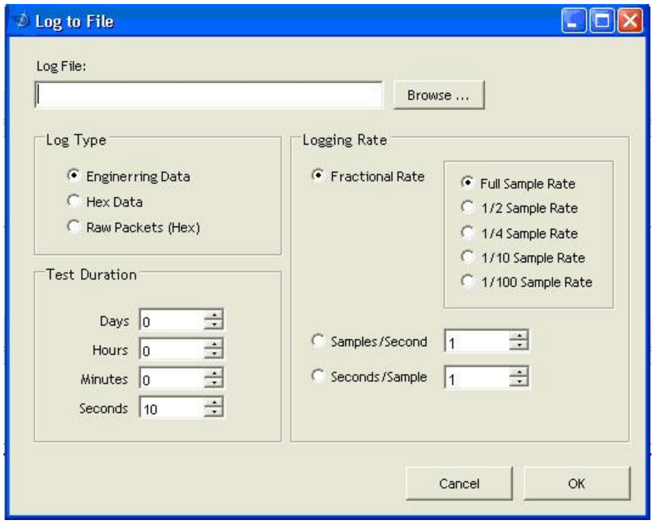


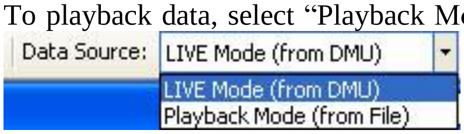
Figure 10: Log to file dialog screen

3. Select the "Browse" box to enter the file name and location that you wish to save your data to.
4. Select the type of data you wish to record. "Engineering Data" records the converted values provided from the system in engineering units, "Hex Data" provides the raw hex values separated into columns displaying the value, and the "Raw Packets" will simply record the raw hex strings as they are sent from the unit.
5. Users can also select a predetermined "Test Duration" from the menu. Using the arrows, simply select the duration of your data recording.
6. Logging Rate can also be adjusted using the features on the right side of the menu.
7. Once you have completed the customization of your data recording, you will be returned to the main screen where you can start the recording process using the button at the top of the page or select "Start Logging" from the "File" menu. Stopping the data recording can be accomplished using the button and the recording can also be paused using the button.

Data playback

In addition to data recording, NAV-VIEW allows the user to replay saved data that has been stored in a log file.

1. To playback data, select "Playback Mode" from the "Data Source" drop down menu at the top.



2. Selecting Playback mode will open a text prompt which will allow users to specify the location of the file they wish to play back. All three file formats are supported (Engineering, Hex, and Raw) for playback. In addition, each time recording is stopped/started a new section is created. These

sections can be individually played back by using the drop-down menu and associated VCR controls.

3. Once the file is selected, users can utilize the VCR style controls at the top of the page to start, stop, and pause the playback of the data.
4. NAV-VIEW also provides users with the ability to alter the start time for data playback. Using the  slide bar at the top of the page users can adjust the starting time.

Raw data console

NAV-VIEW offers some unique debugging tools that may assist programmers in the development process. One such tool is the Raw Data Console. From the "View" drop down menu, simply select the "Raw Data Console". This console provides users with a simple display of the packets that have been transmitted to the unit (Tx) and the messages received (Rx). An example is provided in Figure 11.

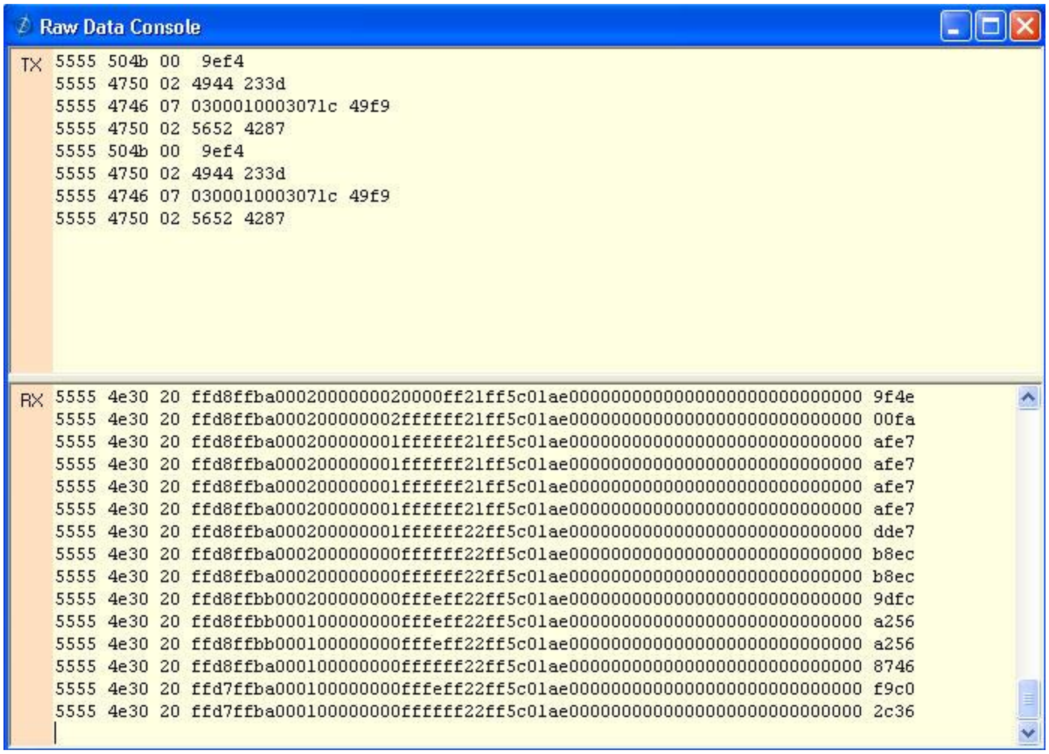


Figure 11: Raw data console

Packet statistics view

Packet statistics can be obtained from the “View” menu by selecting the “Packet Statistics” option (See Figure 12). This view simply provides the user with a short list of vital statistics (including Packet Rate, CRC Failures, and overall Elapsed Time) that are calculated over a one second window. This tool should be used to gather information regarding the overall health of the user configuration. Incorrectly configured communication settings can result in a large number of CRC Failures and poor data transfer.

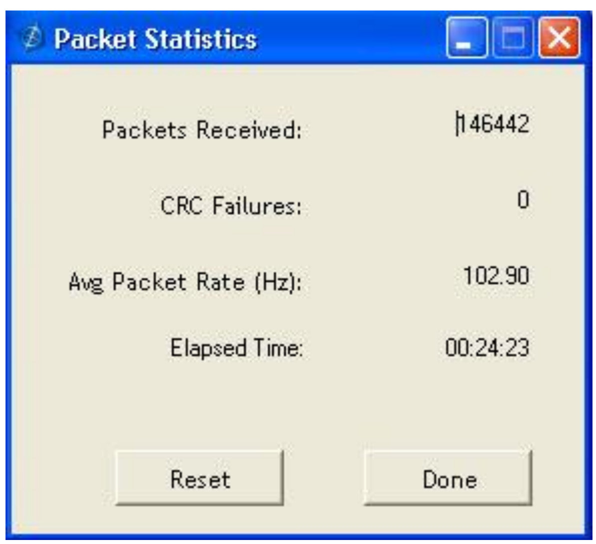


Figure 12: Packet statistics

Unit configuration

The Unit Configuration window (See Figure 13) gives the user the ability to view and alter the system settings. This window is accessed through the "Unit Configuration" menu item under the configuration menu. Under the "General" tab, users have the ability to verify the current configuration by selecting the "Get All Values" button. This button simply provides users with the currently set configuration of the unit and displays the values in the left column of boxes.

There are three tabs within the "Unit Configuration" menu; General, Advanced and BIT Configuration. The General tab displays some of the most commonly used settings. The Advanced and BIT Configuration menus provide users with more detailed setting information that they can tailor to meet their specific needs.

To alter a setting, simply select the check box on the left of the value that you wish to modify and then select the value using the drop-down menu on the right side. Once you have selected the appropriate value, these settings can be set temporarily or permanently (a software reset or power cycle is required for the changes to take affect) by selecting from the choices at the bottom of the dialog box. Once the settings have been altered a "Success" box will appear at the bottom of the page.

IMPORTANT
Caution must be taken to ensure that the settings selected are compatible with the system that is being configured. In most cases a "FAIL" message will appear if incompatible selections are made by the user, however it is the user's responsibility to ensure proper configuration of the unit.

IMPORTANT
Unit orientation selections must conform to the right-hand coordinate system as noted in Section 3.1 of this user manual. Selecting orientations that do not conform to these criteria are not allowed.

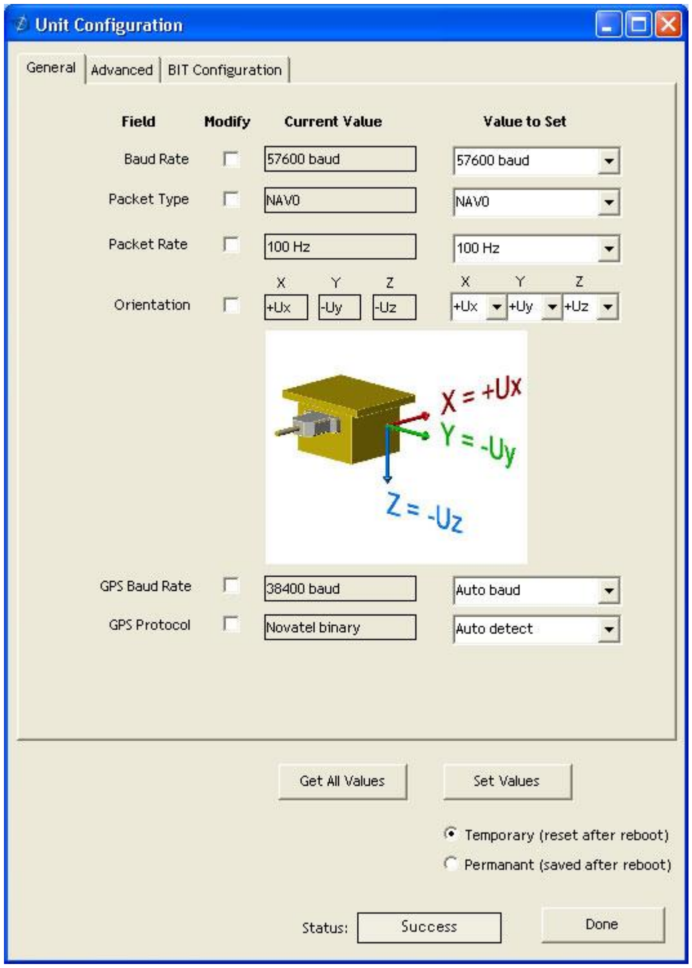


Figure 13: Unit configuration

Advanced configuration

Users who wish to access some of the more advanced features of NAV-VIEW and the INS330BI Series products can do so by selecting the "Advanced" tab at the top of the "Unit Configuration" window.

⚠ WARNING

Users are strongly encouraged to read and thoroughly understand the consequences of altering the settings in the "Advanced" tab before making changes to the unit configuration. These settings are discussed in detail in Chapter 4 below.

Behavior switches (unused with IMU) are identified at the top of the page with marked boxes. A blue box will appear if a switch has been enabled similar to Figure 14 below. The values can be set in the same manner as noted in the previous section. To set a value, users select the appropriate "Modify" checkbox on the left side of the menu and select or enable the appropriate value they wish to set. At the bottom of the page, users have the option of temporarily or permanently setting values. When all selections have been finalized, simply press the "Set Values" button to change the selected settings.

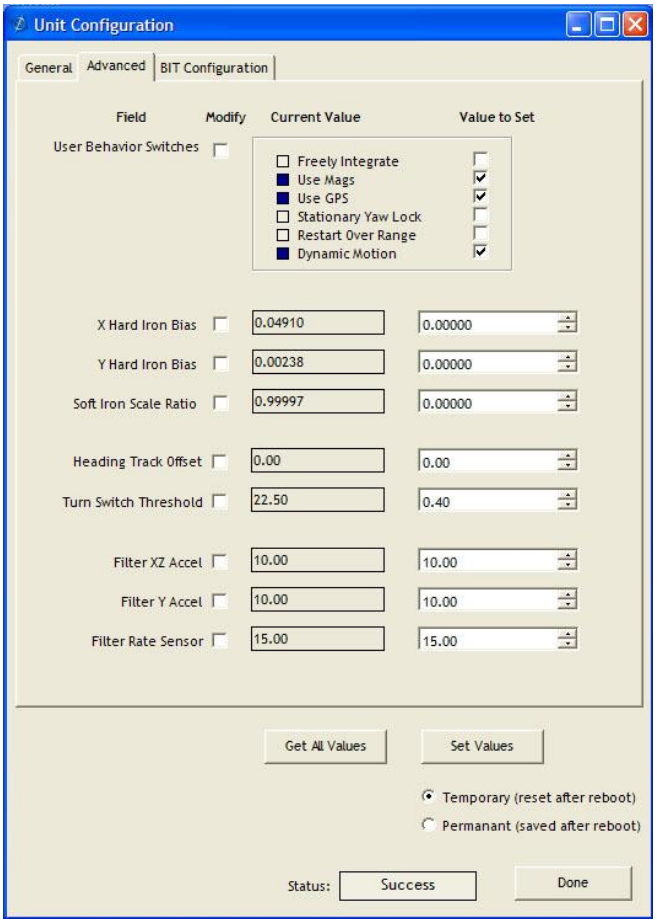


Figure 14: Advanced settings

Read unit configuration

NAV-VIEW allows users to view the current settings and calibration data for a given INS330BI Series unit by accessing the "Read Configuration" selection from the "Configuration" drop down menu (See Figure 15). From this dialog, users can print a copy of the unit's current configuration and calibration values with the click of a button. Simply select the "Read" button at the top of the dialog box and upon completion select the "Print" or "Print Preview" buttons to print a copy to your local network printer. This information can be helpful when storing hard copies of unit configuration, replicating the original data sheet and for troubleshooting if you need to contact ACEINNA's Support Staff.

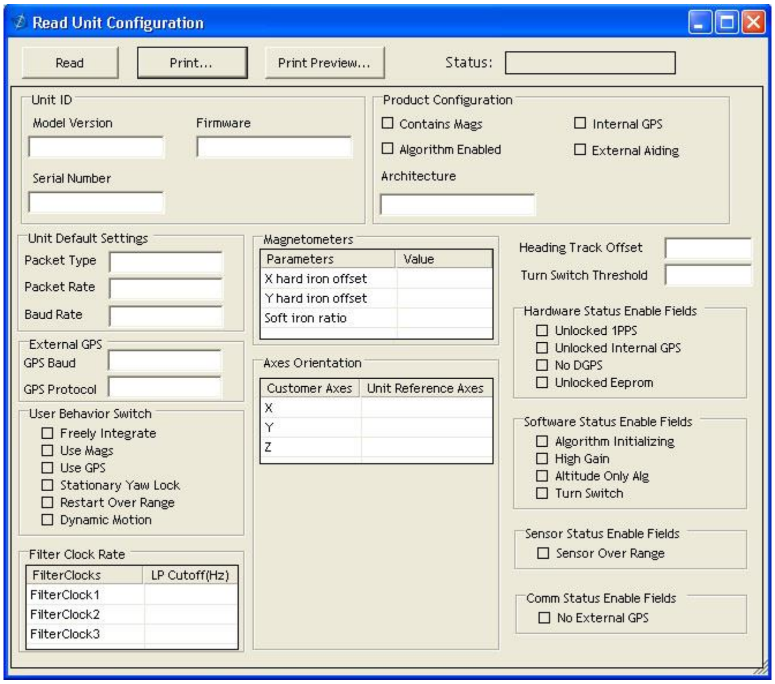
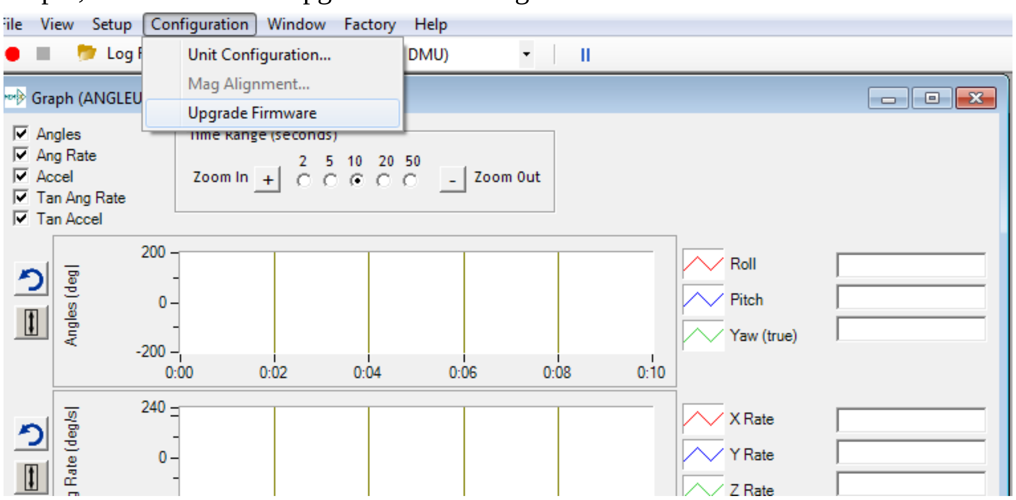


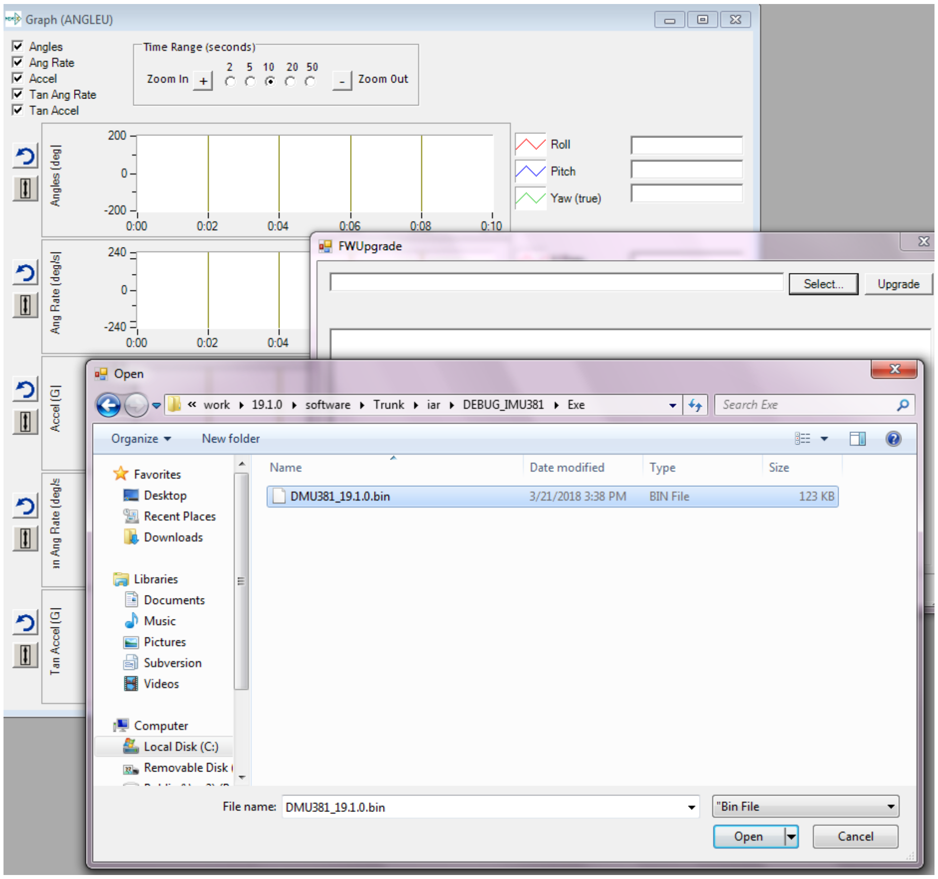
Figure 15: Read configuration

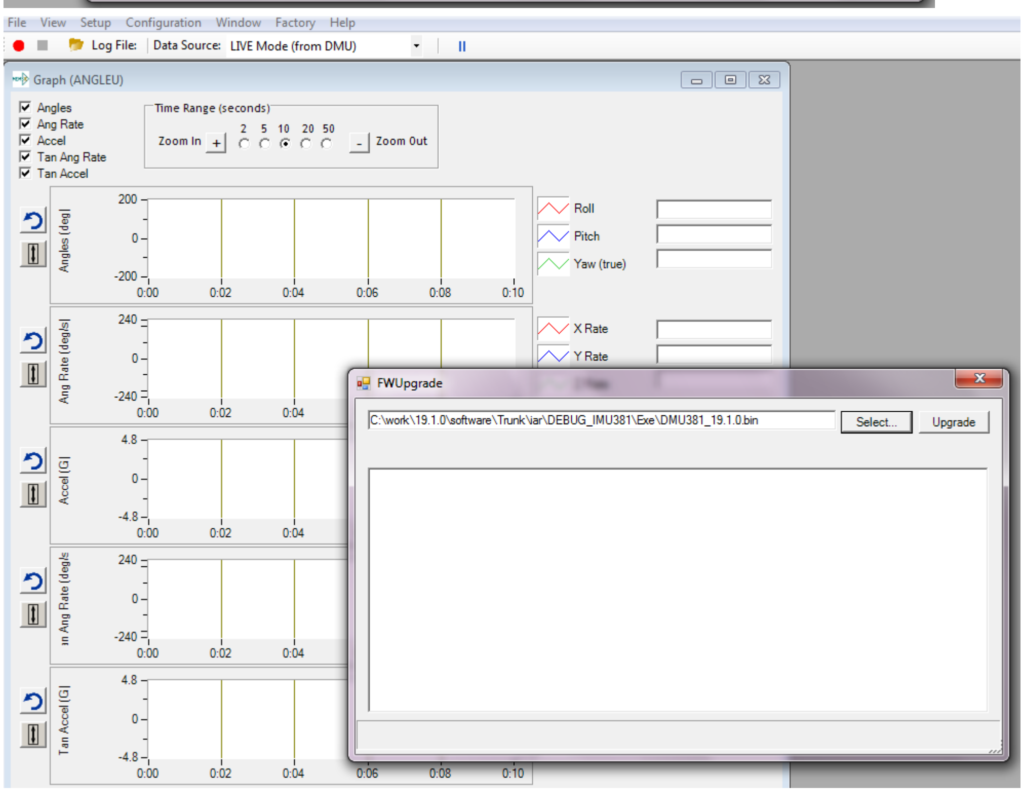
Appendix A: Firmware Upgrade

Step 1, select Firmware upgrade from configuration menu.



Step 2, On pop-up window, select a new version binary file by clicking the SELECT button, then click the Upgrade button.


Graph (ANGLEU)
Angles
Ang Rate
Accel
Tan Ang Rate
Tan Accel
Time Range (seconds)
Zoom In
Zoom Out
Roll
Pitch
Yaw (true)
FWUpgrade
Select...
Upgrade
Open
work › 19.1.0 › software › Trunk › iar › DEBUG_IMU381 › Exe
Search Exe
Organize
New folder
Name
Date modified
Type
Size
DMU381_19.1.0.bin
3/21/2018 3:38 PM
BIN File
123 KB
Favorites
Desktop
Recent Places
Downloads
Libraries
Documents
Music
Pictures
Subversion
Videos
Computer
Local Disk (C:)
File name:
DMU381_19.1.0.bin
"Bin File
Open
Cancel


File View Setup Configuration Window Factory Help
Log File:
Data Source:
LIVE Mode (from DMU)
Graph (ANGLEU)
Angles
Ang Rate
Accel
Tan Ang Rate
Tan Accel
Time Range (seconds)
Zoom In
Zoom Out
Roll
Pitch
Yaw (true)
X Rate
Y Rate
FWUpgrade
C:\work\19.1.0\software\Trunk\iar\DEBUG_IMU381\Exe\DMU381_19.1.0.bin
Select...
Upgrade

Step 3, wait for the process to complee until a success or failure message pops up.

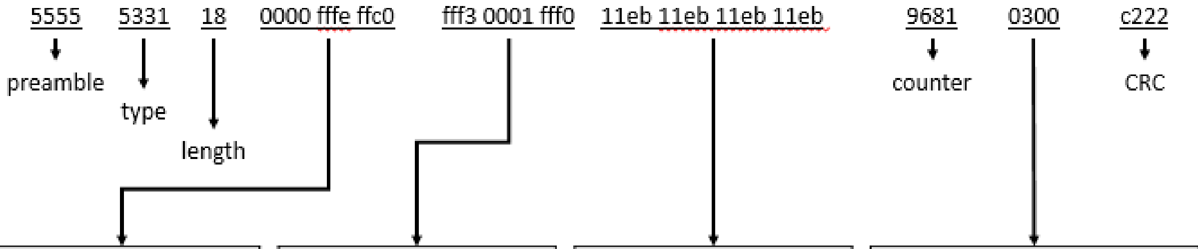
Appendix B: CRC calculation for serial port messages

```
/*******************************************************************************
 * FUNCTION: calcCRC calculates a 2-byte CRC on serial data using
 *     CRC-CCITT 16-bit standard maintained by the ITU
 *                         (International Telecommunications Union).
 * ARGUMENTS: queue_ptr is pointer to queue holding area to be CRCed
 *                         startIndex is offset into buffer where to begin CRC calculation
 *                         num is offset into buffer where to stop CRC calculation
 * RETURNS:          2-byte CRC
 *******************************************************************************/
unsigned short calcCRC(QUEUE_TYPE *queue_ptr, unsigned int startIndex, unsigned int num) {
    unsigned int i=0, j=0;
    unsigned short crc=0x1D0F; //non-augmented initial value equivalent to augmented initial value 0xFFFF

    for (i=0; i<num; i+=1) {
        crc ^= peekByte(queue_ptr, startIndex+i) << 8;

        for(j=0;j<8;j+=1) {
            if(crc & 0x8000) crc = (crc << 1) ^ 0x1021;
            else crc = crc << 1;
        }
    }
    return crc;
}
*******************************************************************************/
```

Appendix C: Sample Packet Decoding



Accelerometers	
Data	Value (g)
0x0000	0.000
0xfffe	-0.0019
0xffc0	-1.000

Angular Rate	
Data	Value (°/s)
0xfff3	-0.8125
0x0001	0.065
0xfff0	-1.000

Temperature	
Data	Value (°C)
0x11eb	28.0
0x11eb	28.0
0x11eb	28.0
0x11eb	28.0

Bit status Field	
Field	Value
masterFail	0
hardwareError	0
reserved	0
softwareError	0
reserved	0000
masterStatus	1
hardwareStatus	1
reserved	0
softwareStatus	0
sensorStatus	0
reserved	000

Figure 16: Example payload from Scaled Data Packet 1 (S1)