IM-8084

Integration Guide for Lodestar and SPRINT Products

Issue A Rev 1

Issue Date: 28 June 2018

Sonardyne International Limited
Blackbushe Business Park
Yateley, Hampshire
GU46 6GD United Kingdom

T. +44 (0) 1252 872288
F. +44 (0) 1252 876100
E. support@sonardyne.com
W. www.sonardyne.com

920-0029

Third party software used with this product is listed in *Appendix A.*

Contacting the Sonardyne Support Team

24-hour Emergency Telephone Helpline: +44 (0) 1252 877600

The Sonardyne 24-hour helpline is answered at the UK Headquarters during normal office hours (08:00 to 17:00). Outside these hours, your call is automatically transferred to an agency, which logs the details of your emergency and alerts the appropriate Sonardyne personnel.

Our aim is to make sure emergency requests are dealt with immediately during office hours, and are responded to within 30 minutes at all other times.

Please note the helpline is for emergency use only.

If you require non-emergency product support, please contact your nearest Sonardyne office. Alternatively, contact the Sonardyne Head Office:

Sonardyne International Ltd
Blackbushe Business Park
Yateley
Hampshire
GU46 6GD
United Kingdom

Telephone: +44 (0) 1252 872288

Fax: +44 (0) 1252 876100

Email: support@sonardyne.com

Note

Email and telephone support is available during normal UK office hours (08:00 to 17:00).

Contents

Contacting the Sonardyne Support Team iii

Amendment History xii

Section 1 – Introduction 1

1.1 Scope of this Manual 1
1.1 Purpose of this Manual 1
1.2 Related Publications 1
1.3 Conventions 1
1.3.1 Formatting Conventions 1
1.3.2 General Conventions 2

Section 2 – Safety 3

2.1 Introduction 3
2.2 Safety Procedures 3
2.2.1 Warnings 3
2.2.2 Cautions 4

Section 3 – Hardware Overview 5

3.1 Hardware Overview 5
3.2 Power 5
3.2.1 Inrush Current 5
3.2.2 Power Pass Through 5
3.3 Serial Communications 6
3.4 Triggers 7
3.5 Ethernet 7
3.6 Common Hardware Versions 7
3.6.1 Lodestar 200 4K 7
3.6.2 Lodestar 300, 500 & 700 4K 9
3.6.3 Detailed Pin Outs 10
3.6.4 SPRINT 300, 500 & 700 4K 11
3.6.5 SPRINT 300, 500 & 700 6K 12
3.6.6 Lodestar/SPRINT 300, 500 & 700 OEM 13
3.6.7 Detailed Pinouts 14
3.6.8 SPRINT-Nav 300, 500 & 700 4K 16
3.6.9 SPRINT-Nav 300, 500, & 700 6K 19
3.6.10 Detailed Pin Outs 21

Section 4 – Lodestar/SPRINT Concepts and Definitions 23

4.1 Time 23
4.1.1 Instrument Time (System Time) 23
4.1.2 Common Time (UTC) 23
4.2 Frames 23
4.2.1 IMU Body Frame 23
4.2.2 Vehicle 23
4.2.3 Navigation (NED) 23

4.3 Lever Arms and Mounting Angles 23
4.3.1 Lever Arms 23
4.3.2 Mounting Angles 24
4.4 Coordinate System 25
4.5 Navigation Algorithms 25
4.5.1 Inertial Measurement Unit (IMU) 25
4.5.2 Attitude and Heading Reference System (AHRS) 26
4.5.3 Inertial Navigation System (INS) 26
4.6 Aiding Sources 27
4.6.1 External Position Aiding (XPOS) 27
4.6.2 USBL Aiding (SUSBL) 27
4.6.3 GNSS Aiding (GPS) 27
4.6.4 LBL 28
4.6.5 Zero Velocity Update Aiding (ZUPT) 28
4.6.6 Zero Mean Depth (ZMD) 28
4.6.7 Vehicle-Mounted Sensor Aiding 28
Section 5 – Protocol and Messaging Overview 29
5.1 Multiplex Format 29
5.1.1 Byte Stuffing 29
5.1.2 Packet Format 29
5.1.3 Code Example 31
5.2 Inputs, Outputs and Logs 32
5.2.1 Typical Input Processing 32
5.2.2 Output Messages 32
5.2.3 Log Messages 33
Section 6 – Command and Control Overview 35
6.1 Port Commands 35
6.2 Non-Multiplex 35
6.3 Multiplex 35
6.4 Command Types 35
6.4.1 Status Commands 36
6.4.2 Configuration Commands 36
6.4.3 Macros 37
6.5 Complete Lodestar/SPRINT Configuration 38
Section 7 – Step-by-Step Integration (Example) 39
7.1 Initial Overview 39
7.1.1 Normal Boot Sequence 40
7.1.2 Resetting to Factory default 40
7.2 Initial Configuration 41
7.3 Multiplex AHRS Message Parsing 41
7.4 Monitoring 43
7.4.1 BIST 43
7.4.2 Observation Status Messages 44

7.4.3 Time System Status ... 44
7.4.4 General Logging ... 44
7.5 Multiplex Input Message Aiding ... 45
7.5.1 Time ... 45
7.5.2 Position/Depth/Velocity ... 48
7.6 Lodestar/SPRINT Configuration ... 68
7.6.1 IMU Configuration ... 68
7.6.2 GPS Aiding ... 69
7.6.3 XPOS Aiding ... 70
7.6.4 SUSBL Aiding ... 71
7.6.5 LBL Aiding ... 74
7.6.6 PRESS (Pressure Depth) Aiding ... 75
7.6.7 ZMD Aiding ... 76
7.6.8 DVL Aiding ... 76
7.6.9 SVS Aiding ... 79
7.6.10 ZUPT Aiding ... 80
7.6.11 INS Configuration ... 80
7.6.12 Trigger Configuration ... 85
7.6.13 Time System Configuration ... 86
7.7 Multiplex Command and Control ... 87
7.8 Multiplex Output and Log Messages ... 89
7.8.1 INS Binary Message (LNAV) ... 89
7.8.2 Remote Point $PSONNAV ... 92
7.8.3 BIST ... 93
7.8.4 Settings ... 95
7.9 Ethernet Connectivity ... 97
7.9.1 Ethernet Input ... 97
7.9.2 Ethernet Outputs and Logging ... 97
7.10 Integrated Instrument Addendum ... 97
7.10.1 Combined Units Lodestar-Nav and SPRINT-Nav ... 97
7.11 Operational Considerations ... 98
7.11.1 Establishing a Connection to Lodestar/SPRINT ... 98
7.11.2 Low Latency Outputs ... 99
7.11.3 Aiding Switching ... 99
7.11.4 Power Pass Through ... 100
7.11.5 Communications Pass Through ... 100
7.11.6 Providing Time Synchronisation from Lodestar/SPRINT ... 103
7.11.7 Communication and CPU Loading Checks ... 103
7.11.8 DVL Calibration ... 103
7.11.9 IMU Output ... 104
Section 8 – Messages ... 105
8.1 Message IDentifiers (MIDs) ... 105
8.2 IN Messages ... 107

8.2.1 Digiquartz Pressure Sensor Report 107
8.2.2 $PSONDEP Report 107
8.2.3 $__DPT REPORT 108
8.2.4 Tritech Winson Processed Data 109
8.2.5 Midas SVX2 Depth 110
8.2.6 XDEPTH 111
8.2.7 PD4 112
8.2.8 PD0 114
8.2.9 Checksum Data Format 127
8.2.10 $_ZDA Report 127
8.2.11 $_GGA Report 128
8.2.12 VTG 129
8.2.13 PSONLOBS 130
8.2.14 PSONLBLBCN 131
8.2.15 Simrad SSB – SSBL Position Report ($PSIMSSB) 131
8.2.16 $PSONSS Report 133
8.2.17 Valeport Sensor Telegram 133
8.2.18 XPOS 134
8.3 LOG/OP Messages 135
8.3.1 CPU, UART & PWRSTAT 135
8.3.2 Observation Status Message 135
8.3.3 LSZDA 144
8.3.4 PRDKELLBIN 144
8.3.5 Time System Data (TMS) 145
8.3.6 BIST 146
8.3.7 TRG 151
8.3.8 NAVCAL 152
8.3.9 Navigation Quality Estimate (NavQual) 152
8.3.10 Navigation Data (Nav) 153
8.3.11 PRDID 155
8.3.12 TSS1 156
8.3.13 TSS2 157
8.3.14 TSS3 158
8.3.15 EM1000 160
8.3.16 EM3000 161
8.3.17 PHTRO 162
8.3.18 HDT 163
8.3.19 THS 164
8.3.20 TEMP 165
8.3.21 SON2 166
8.3.22 POSMV111 167
8.3.23 POSMV113 169
8.3.24 PSONNAV 170

8.3.25 LNAV & LNAVUTC 171
Section 9 – Command and Control 175
9.1 SYS Commands 175
9.2 GC Commands 175
9.3 OP Commands 176
9.4 IN Commands 176
9.5 PORT Commands 176
9.6 INS Commands 177
9.7 GPS Commands 177
9.8 SUSBL Commands 177
9.9 LBL Commands 178
9.10 XPOS Commands 178
9.11 ZMD Commands 178
9.12 SVS Commands 178
9.13 PRESS Commands 179
9.14 DVL Commands 179
9.15 ZUPT Commands 179
9.16 TSYS Commands 179
9.17 LOG Commands 180
9.18 TRIG Commands 180
Appendix A – Open Source Software Licence Notices and Terms 181
A.1 Open Source Software 181
Definitions 182

Figures

Figure 3-1 Lodestar/SPRINT Current Profile 5
Figure 3-2 Lodestar 200 4K Outline Drawing 7
Figure 3-3 Lodestar 300/500/700 4K Outline Drawing 9
Figure 3-4 SPRINT 300/500/700 4K Outline Drawing 11
Figure 3-5 SPRINT 300/500/700 6K Outline Drawing 12
Figure 3-6 Lodestar 300/500/700 OEM Outline Drawing 13
Figure 3-7 SPRINT-Nav 300/500/700 4K Outline Drawing 16
Figure 3-8 SPRINT-Nav 300/500/700 6K Outline Drawing 19
Figure 4-1 Lodestar/SPRINT Coordinate Frame 25
Figure 7-1 SUSBL Time of Validity Source Selection 73
Figure 7-2 Output Trigger Configuration Parameters 86
Figure 7-3 Steps for Establishing a Connection 99
Figure 7-4 DVL Calibration Report 104

Tables

Table 1-1 Related Publications ... 1
Table 1-2 Conventions used in this Manual ... 1
Table 3-1 Typical Serial Port Number to Connector Name ... 6
Table 3-2 Typical Trigger Number to Port Name ... 7
Table 3-3 SPRINT-Nav 300, 500 & 700 4K Connector Functions ... 18
Table 3-4 SPRINT CP/E1 Port (16-Pin Seacon) Pinout ... 21
Table 3-5 SPRINT C1 Port (7-Pin Seacon) Pinout ... 22
Table 3-6 SPRINT Transceiver (T1/T2) Port (7-Pin Seacon) Pinout ... 22
Table 3-7 Syrinx DVL Port (16-Pin Seacon) Pinout ... 22
Table 4-1 INS Aiding Sources ... 27
Table 5-1 Packet Start and End Characters ... 29
Table 5-2 Multiplex Packet Format ... 29
Table 5-3 Multiplex ID Field Format ... 30
Table 5-4 LOG message ASCII wrapper format ... 33
Table 6-1 Message Macros ... 37
Table 7-1 SON2 Multiplex Decode ... 42
Table 7-2 ZDA Multiplex Encoding ... 47
Table 7-3 PSIMSSB Multiplex Encoding ... 49
Table 7-4 SUSBL GGA Multiplex Encoding ... 51
Table 7-5 GPS GGA Multiplex Encoding ... 53
Table 7-6 VTG Multiplex Encoding ... 55
Table 7-7 XPOS Multiplex Encoding ... 57
Table 7-8 PSONLOBS Multiplex Encoding ... 59
Table 7-9 PSONLBLBCN Multiplex Encoding ... 59
Table 7-10 Supported PRESS sensor types ... 60
Table 7-11 XDEPTH Multiplex Encoding ... 62
Table 7-12 Supported DVL Messages ... 63
Table 7-13 PSONSS Multiplex Encoding ... 67
Table 7-14 Mounting Angle Input Limits ... 69
Table 7-15 Multiplexed OBSTDVL Message ... 83
Table 7-16 OBSTDVL Message Field Decode ... 84
Table 7-17 OBSTDVL Rejection Bit Field Decode ... 84
Table 7-18 Multiplexed "GC SETTLE" Command ... 88
Table 7-19 Multiplexed Response to "GC SETTLE" Command ... 88
Table 7-20 Multiplexed INS LNAV message ... 90
Table 7-21 INS LNAV Message Decode ... 90
Table 7-22 INS LNAV Message Status Decode ... 91
Table 7-23 Multiplexed PSONNAV Message (based on Remote Point) ... 92
Table 7-24 Multiplexed BIST Message ... 93
Table 7-25 BIST Field Decode ... 94
Table 7-26 Active Bits in Comms Field ... 94
Table 7-27 Multiplexed Setting Message Fragment (First) ... 95

Table 7-28 Multiplexed Setting Message Fragment (Last) ... 96
Table 7-29 Multiplexed Pass Through Message (To Lodestar/SPRINT) ... 101
Table 7-30 Multiplexed Pass Through Message (From Lodestar/SPRINT) ... 101
Table 8-1 Message Identifiers ... 105
Table 8-2 PD4 ... 112
Table 8-3 Generic Observation Status Fields ... 135
Table 8-4 Generic Rejection Bits ... 135
Table 8-5 OBSTZMD Specific Fields ... 136
Table 8-6 OBSTGPSPOS Specific Rejection Bits ... 136
Table 8-7 OBSTGPSPOS Specific Fields ... 137
Table 8-8 OBSTSUSBL Specific Rejection Bits ... 137
Table 8-9 OBSTSUSBL specific fields ... 138
Table 8-10 OBSTXPOS Specific Rejection Bits ... 138
Table 8-11 OBSTXPOS Specific Fields ... 139
Table 8-12 OBSTPDEPTH Specific Rejection Bits ... 139
Table 8-13 OBSTPDEPTH Specific Fields ... 140
Table 8-14 OBSTSVS Specific Rejection Bits ... 140
Table 8-15 OBSTSVS Specific Fields ... 141
Table 8-16 OBSTDVL Specific Rejection Bits ... 141
Table 8-17 OBSTDVL Specific Fields ... 142
Table 8-18 OBSTDVL Beam Rejection Status Fields ... 142
Table 8-19 OBSTLBL Specific Rejection Bits ... 143
Table 8-20 OBSTLBL Specific Fields ... 144
Table 8-21 OBSTZUPT Specific Fields ... 144
Table 8-22 BIST Message Blocks ... 146
Table 8-23 BIST – IMU Block ... 146
Table 8-24 BIST – Comms Block ... 147
Table 8-25 BIST – CCA block ... 149
Table 8-26 BIST – AHRS block ... 150
Table 8-27 BIST – AINS block ... 150
Table 8-28 Navigation Data ... 153
Table 8-29 PRDID Formatting ... 155
Table 8-30 TSS1 Formatting ... 156
Table 8-31 TSS2 Formatting ... 157
Table 8-32 TSS3 Formatting ... 158
Table 8-33 EM1000 Formatting ... 160
Table 8-34 EM3000 Formatting ... 161
Table 8-35 PHTRO Formatting ... 162
Table 8-36 HDT Formatting ... 163
Table 8-37 THS Formatting ... 164
Table 8-38 TEMP Formatting ... 165
Table 8-39 SON2 Formatting ... 166
Table 8-40 POSMV111 Formatting ... 167

Table 8-41 POSMV113 Formatting 169
Table 8-42 PSONNAV Formatting 170
Table 8-43 LNav Data 171
Table 8-44 LNav Status 174

Amendment History

The amendment history records all amendments and additions made to this manual.

Issue	Revision	Date	Comments	Section	Page
A	0	22/06/2017	Draft	All	All
A	1	28/06/2018	First Issue	All	All

Section 1 – Introduction

1.1 Scope of this Manual

This manual provides the information required by an integrator to integrate a Lodestar based product onto their vehicle/platform.

Lodestar and SPRINT can be used for a variety of applications. For the majority of these applications Sonardyne's Lodestar PC Application and/or SPRINT PC Application can be used to configure the instrument and set up any outputs. This manual covers applications that either have more complex input/output requirements in conjunction with the PC Application(s) or where the PC Application(s) are being wholly replaced by an end user's system, for example ROV control or AUV system integration.

Note

"Lodestar Based Products". Lodestar is Sonardyne's Gyrocompass and Motion sensor product. The same hardware and majority of firmware are also used in SPRINT based products. Data formats and configuration formats are shared between the two product lines. Features that are only available on SPRINT products will be acknowledged in the manual.

1.1 Purpose of this Manual

This manual is intended for manufacturers and operators who wish to integrate Lodestar and SPRINT products into their vehicles/platforms.

To make sure the safety of the installer and operator is maintained it is important that all warnings and cautions in *Section 2 –Safety* of this manual and in any related manuals are read and fully understood.

1.2 Related Publications

To make sure the equipment is operated safely, a *Sonardyne Safety Manual* is supplied with this user manual. It is important that the *Sonardyne Safety Manual* is read and fully understood before proceeding with any activity on the equipment.

Table 1-1 Related Publications

Publication	Title
Sonardyne Safety Manual	*Operational and Safety Precautions*
Kongsberg APOS	*Release 4.2.2 Manual (29.April. 2005)*

1.3 Conventions

1.3.1 Formatting Conventions

Table 1-2 Conventions used in this Manual

Format	Convention
Italic Type	References to Figures, Tables, Sections and internal/external source

1.3.2 General Conventions

- This document will be supplied together with a number of other files that will help to demonstrate aspects of integrating a Lodestar/SPRINT based product. Where an external file is referenced the path to that file will be stated assuming that this document is being read from its location on the SDK USB Memory Stick.
- Where a number is prepended by “0x” it should be assumed that the following characters are a number in hexadecimal notation. E.g. 0x40 = 64
- Where a number is postpended by “b” it should be assumed that the number is being shown in binary notation with the digit closest to the “b” being the least significant bit. E.g. 11100000b = 224
- Care should be taken when reading message structures or anything else that could be held in a software array. Depending on the source of the data the first byte of data may be numbered 0 or 1.
- Where there are examples of command syntax shown within the document this will be shown in fixed width font and for clarity will not show the required <CR> and <LF> characters, e.g.

  ```
  GC LAT 51.55
  ```

  Should be interpreted as:

  ```
  GC LAT 51.55<cr><lf>
  ```

  Where the direction of the communications is important the following convention will be used at the start of the line in this document (but should not be entered when sending data or be expected when receiving data from a Lodestar/SPRINT):

`>>GC LAT`	This is being sent to the Lodestar/SPRINT
`<<GC LAT 51.55`	This is being received from the Lodestar/SPRINT

- The terms “transponder” and “beacon” are used synonymously in Sonardyne documentation and in the subsea positioning literature in general and are therefore interchangeable.
- Angles are measured and entered in degrees unless otherwise stated.
- Where pseudo code (code snippets/bitwise operations) is provided in this document it will follow a ‘C’ coding style and operator usage

Section 2 – Safety

2.1 Introduction

Before any activity is carried out on the equipment, it is recommended that the included *Sonardyne Safety Manual* and all warnings and cautions in this manual are fully read and understood.

It is recommended that the operator complies with the Health and Safety Regulations applicable to the vessel and the region before operating this equipment.

Operators and service personnel must be familiar with the normal operating and safety procedures for subsea equipment.

Documentation must be consulted whenever a ⚠ or ⚠ warning symbol is found on the equipment, this is in order to determine the nature of the potential hazard and any actions which must be taken.

If any additional equipment is used, any warnings and cautions in the equipment user manual must be read and fully understood and the equipment must only be used as specified by the manufacturer.

The safety of any system incorporating this equipment is the responsibility of the assembler of the system.

2.2 Safety Procedures

2.2.1 Warnings

⚠ **Personal protection. Appropriate protective equipment such as protective footwear, hard hat and gloves must be worn when handling or carrying out any procedures involving Sonardyne and other equipment**

⚠ **Heavy equipment. Many Sonardyne products and equipment types, such as SPRINT Unit, transponders, transceivers, cable drums etc. require Manual Handling Equipment (MHE) for lifting due to their heavy weight. If MHE is not available, it is the responsibility of the operator to perform a manual handling risk assessment prior to carrying out manual lifting/handling. Refer to the individual equipment documentation for weight specifications.**

⚠ **Dismanting. This instrument must only be accessed internally and dismantled by qualified Sonardyne personnel.**

⚠ **Lithium-ion Battery Pack. This instrument contains a backup lithium-ion battery pack. Refer to the Sonardyne Safety Manual for saftey information for lithium-ion batteries.**

⚠ **Risk of toxic gases and Corrosive Liquids. Do not stand in direct line with the end of the unit when operating the Pressure Relief Vent Valve. Sudden release of high pressure gases could cause injury to personnel. Wear Personal Protective Equipment such as goggles when operating the Pressure Relief Vent Valve.**

2.2.2 Cautions

⚠ **Incorrect Power Supply. Make sure the Lodestar/SPRINT is supplied with a suitable DC Voltage (see specification of unit for suitable voltage range). Do not use an AC power supply.**

⚠ **Damage to connectors. Connector caps should be fitted whenever cables are not plugged in. The connectors are dry-mate and the connector faces/pins must not get wet at any time.**

Section 3 – Hardware Overview

3.1 Hardware Overview

This section provides a non-exhaustive description of the Lodestar/SPRINT hardware. It is written from the perspective of the underlying "generic" hardware capability; this may be different from what is available externally from the instrument, depending on connector choice.

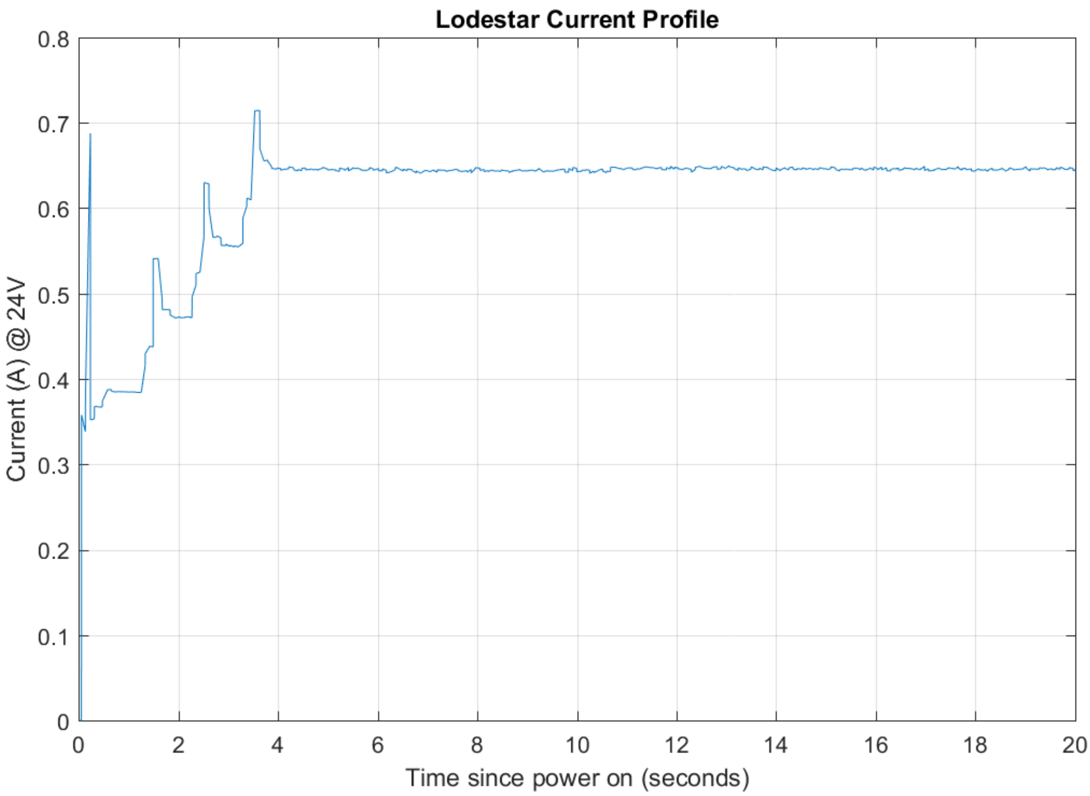
3.2 Power

A generic Lodestar/SPRINT will work with an input DC power supply between 20–50 V. However some instruments in the range, e.g. SPRINT-Nav have requirements that stipulate a 24 V supply.

3.2.1 Inrush Current

Figure 3-1 shows the current profile for a standard Lodestar/SPRINT powered with a 24 V dc supply. The power consumption (i.e. watts) does not change with input voltage.

Figure 3-1 Lodestar/SPRINT Current Profile



3.2.2 Power Pass Through

Up to three ports may offer the feature of "Power Pass Through". This allows the supply power to the Lodestar/SPRINT to be passed through (routed) to another port. This feature is generally used to power another instrument (Sound Speed, DVL, Pressure Sensor, etc.) that is also connected to the serial communications provided by that port. For most instruments this means that only one connection is required (supplying both power and Serial communications).

By design the power pass through on all ports is off by default and will return to this state after the instrument is reset. Enabling of power pass through is via command.

Notes

No voltage level translation takes place, therefore the system integrator/user must ensure that anything powered using the power pass through is capable of being powered by the main Lodestar/SPRINT power supply.

Very little conditioning takes place on the pass through supply output, therefore any noise or brown-out will also likely be passed through to the connected instruments.

If external power is lost and a battery is fitted to the Lodestar/SPRINT, it will only be used to power the Lodestar/SPRINT; it will not supply power for power pass through. In a Lodestar-Nav/SPRINT-Nav the internal pressure transducer will continue to be powered as this is run by the Lodestar/SPRINT internal power rail. The Syrinx DVL will not be powered.

The power pass through ports have short circuit and over-current protection. An output will trip if the current on that port exceeds 3 A. The port must be turned off and back on via command to reset the trip. Lodestar functionality including other power pass through ports is not affected if one port trips.

3.3 Serial Communications

The generic Lodestar/SPRINT units have a total of five Universal Asynchronous Receivers/Transmitters (UARTs). These are numbered 0–4, but will usually be identified by the connector identification of the housing; see *Table 3-1*.

The UARTS can be configured to the following modes:

- Baud Rate: 4800, 9600, 19200, 38400, 57600, 115200, 230400, 921600
- Stop Bit: 1 or 2 bits
- Parity: Odd, Even and None
- Data: 8 or 7 Bit

Depending on the connector type and port used, the port may support RS232, RS485 Full Duplex or RS485 Half Duplex.

Table 3-1 Typical Serial Port Number to Connector Name

Port Number	Typical Connector Identification Name	Typical Operating Modes
0	CP (Console Port)	RS232 or RS485 Full Duplex (externally selectable by pin)
1	C1	RS232 only
2	C2	RS232 or RS485 Half Duplex
3	T1	RS232 or RS485 Half Duplex
4	T2	RS232 or RS485 Half Duplex

Note

Port C2 is designed for internal use only and is not isolated. As such it is not usually exposed as an external connector in Sonardyne's product range.

3.4 Triggers

The generic Lodestar/SPRINT supports two output and two input triggers. Similar to the UARTs, these are numbered 1–4, but will usually be identified by the connector identification of the housing. A typical mapping of trigger numbers to connector identifications is shown in *Table 3-2*.

Table 3-2 Typical Trigger Number to Port Name

Trigger Number	Typical Connector Identification Name	Trigger Type
1	C1	Input
2	E1	Input
3	T1	Output
4	T2	Output

3.5 Ethernet

Lodestar/SPRINT provides one Ethernet port with a recommended connection speed of 100 Mbits.

Note

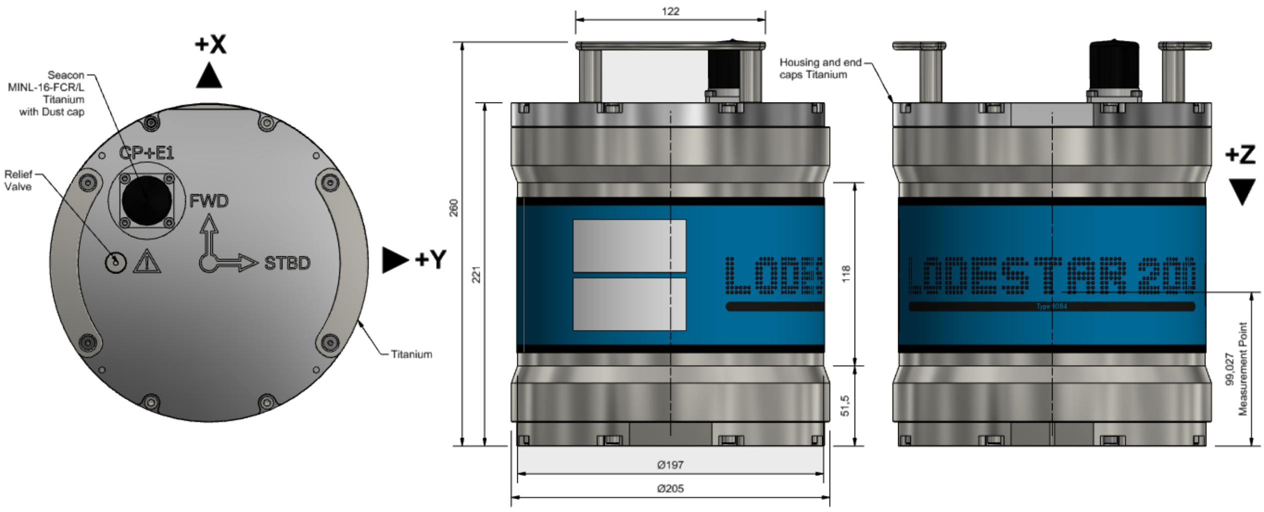
By default, the IP address of a Lodestar/SPRINT is 192.168.179.50. This can be changed via command and control. There is no DHCP client support.

3.6 Common Hardware Versions

Unless otherwise stated any measurements provided in this section are in millimetres (mm).

3.6.1 Lodestar 200 4K

Figure 3-2 Lodestar 200 4K Outline Drawing



3.6.1.1 Connector Pinouts

Port	Connectors			
	Type	Internal Sensor	Detailed Pinout (section)	Function
CP/E1	Seacon MINL-16FCR/L	SPRINT INS	*Section 3.6.1.5*	RS232 and RS485 Full Duplex Communications and Input Power Ethernet (100 Mbit/s) Communications and Input Trigger

3.6.1.2 Power

Input: 20–50 V dc, 15W nominal, 35 W max

3.6.1.3 Shock Vibration and Temperature

Operating Temperature: -20 to +55°C

Storage Temperature: -20 to +60°C

Shock Rating (Operational): 22 g, 11 ms half sine.

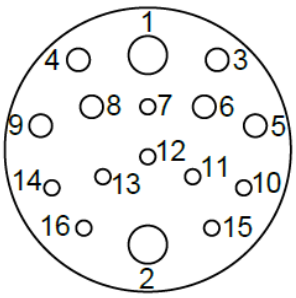
3.6.1.4 Weight

In Air: 18.5 Kg

In Water: 11.5Kg

3.6.1.5 Detailed Pin Outs

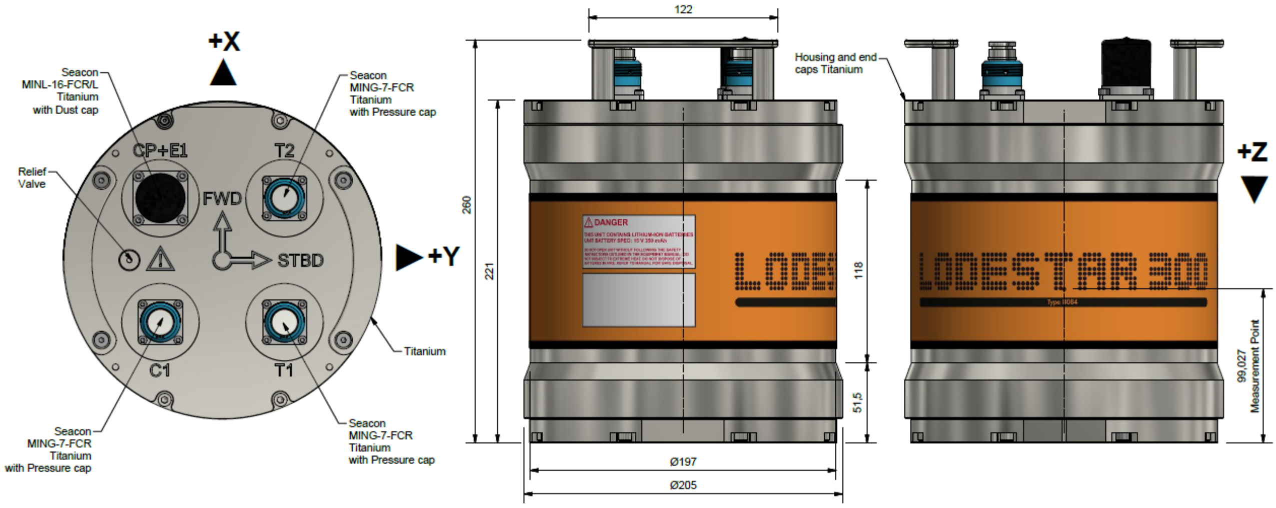
Face view



Seacon Pin No.	Function
1	DC 0 V
2	DC In
3	Comms/Trig Ground
4	Screen
6	Trigger In
7	Ethernet TD -
8	RS232/485 Select **Connect to 0 V1/Pin3 for RS232** **Do not connect for RS485**
10	Ethernet RD -
11	RX TX+
12	Ethernet TD +
13	TX TX -
14	RX -
15	Ethernet RD +
16	RX +

3.6.2 Lodestar 300, 500 & 700 4K

Figure 3-3 Lodestar 300/500/700 4K Outline Drawing



3.6.2.1 Connector Pinouts

Port	Connectors			
	Type	Internal Sensor	Detailed Pinout (section)	Function
CP/E1	Seacon MINL-16FCR/L	SPRINT INS	*Section 3.6.3.1*	RS232 and RS485 Full Duplex Communications and Input Power Ethernet (100 Mbit/s) Communications and Input Trigger
C1	Seacon MING-7-FCR	SPRINT INS	*Section 3.6.3.2*	RS232 Communications, Input Trigger and Power Pass Through
T1	Seacon MING-7-FCR	SPRINT INS	*Section 3.6.3.3*	RS232 and RS485 Half Duplex Communications, Output Trigger and Power Pass Through
T2	Seacon MING-7-FCR	SPRINT INS	*Section 3.6.3.3*	RS232 and RS485 Half Duplex Communications, Output Trigger and Power Pass Through

3.6.2.2 Power

Input: 20–50 V dc, 15 W nominal, 35 W max

3.6.2.3 Shock Vibration and Temperature

Operating Temperature: -20 to +55°C

Storage Temperature: -20 to +60°C

Shock Rating (Operational): 22 g, 11 ms half sine.

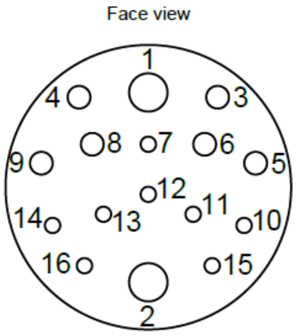
3.6.2.4 Weight

In Air: 18.5 Kg

In Water: 11.5 Kg

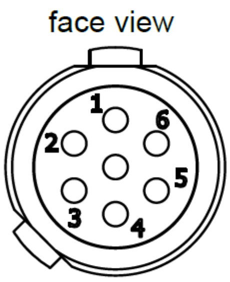
3.6.3 Detailed Pin Outs

3.6.3.1 SPRINT CP/E1 Port (16 Pin Seacon)



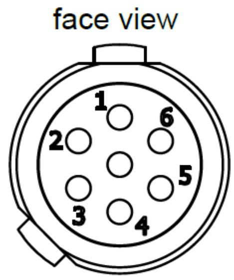
Seacon Pin No.	Function
1	DC 0 V
2	DC In
3	Comms/Trig Ground
4	Screen
6	Trigger In
7	Ethernet TD -
8	RS232/485 Select **Connect to 0 V1/Pin3 for RS232** **Do not connect for RS485**
10	Ethernet RD -
11	RX TX+
12	Ethernet TD +
13	TX TX -
14	RX -
15	Ethernet RD +
16	RX +

3.6.3.2 SPRINT C1 Port (7 Pin Seacon)



Seacon Pin No.	Function
2	Comms / Trig Ground
3	Trigger In
4	DC 0 V
5	TX TX-
6	RX TX+
7	DC Out

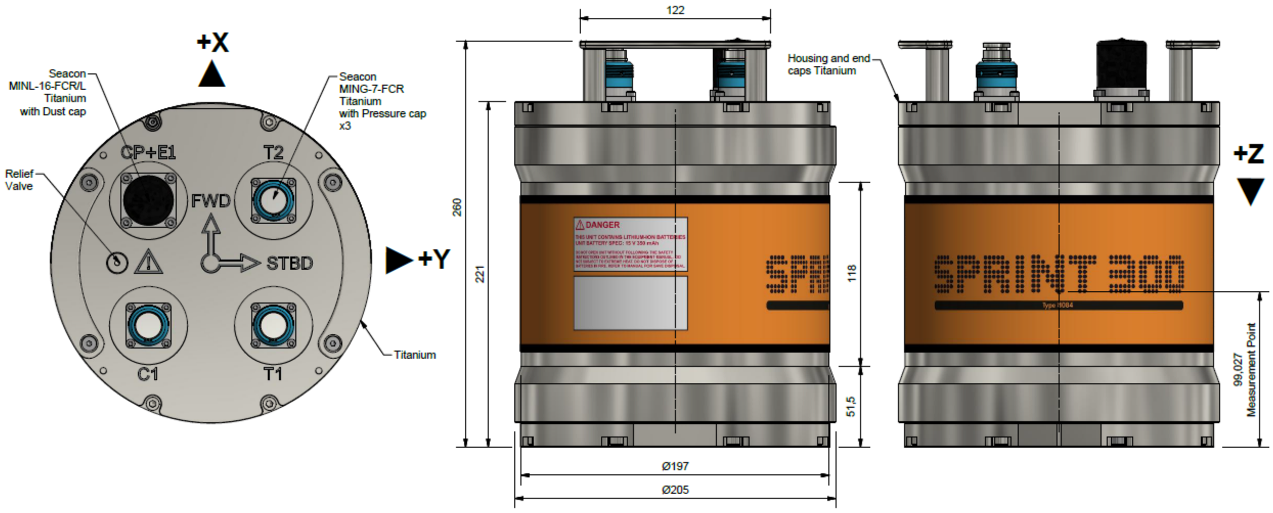
3.6.3.3 SPRINT Transceiver (T1 / T2) Port (7 Pin Seacon)



Seacon Pin No.	Function
2	Comms / Trig Ground
3	Trigger Out
4	DC 0 V
5	TX TX-
6	RX TX+
7	DC Out

3.6.4 SPRINT 300, 500 & 700 4K

Figure 3-4 SPRINT 300/500/700 4K Outline Drawing



3.6.4.1 Connector Pinouts

The connector pinouts are the same for the Lodestar/SPRINT 300, 500 & 700 4K and 6K; see *Section 3.6.2.1*.

3.6.4.2 Power

The power requirements are the same for the Lodestar/SPRINT 300, 500 & 700 4K and 6K; see *Section 3.6.2.2.*

3.6.4.3 Shock Vibration and Temperature

The environmental specifications are the same for the Lodestar/SPRINT 300, 500 & 700 4K and 6K; see *Section 3.6.2.3*.

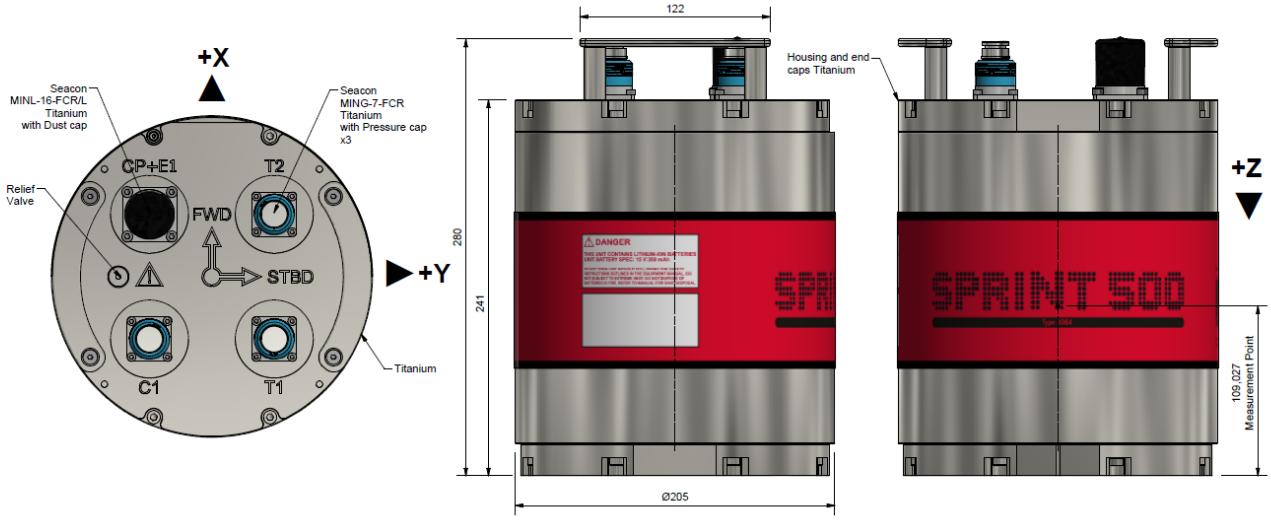
3.6.4.4 Weight

In Air: 18.5 Kg

In Water: 11.5 Kg

3.6.5 SPRINT 300, 500 & 700 6K

Figure 3-5 SPRINT 300/500/700 6K Outline Drawing



3.6.5.1 Connector Pinouts

The connector pinouts are the same for the Lodestar/SPRINT 300, 500 & 700 4K and 6K; see *Section 3.6.2.1*.

3.6.5.2 Power

The power requirements are the same for the Lodestar/SPRINT 300, 500 & 700 4K and 6K; see *Section 3.6.2.2.*

3.6.5.3 Shock Vibration and Temperature

The environmental specifications are the same for the Lodestar/SPRINT 300, 500 & 700 4K and 6K; see *Section 3.6.2.3*.

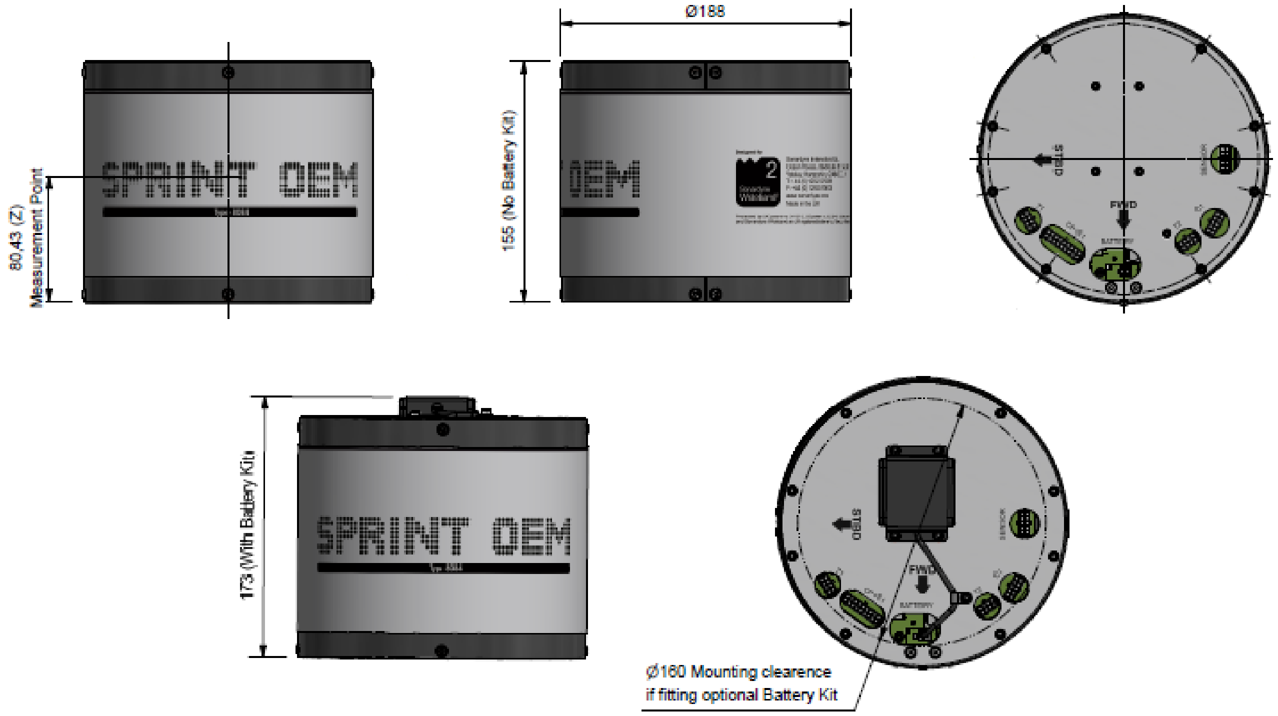
3.6.5.4 Weight

In Air: 22.0 Kg

In Water: 14.0 Kg

3.6.6 Lodestar/SPRINT 300, 500 & 700 OEM

Figure 3-6 Lodestar 300/500/700 OEM Outline Drawing



3.6.6.1 Connector Pinouts

Port	Connectors		
	Type	**Detailed Pinout (Section)**	**Function**
CP/E1	MOLEX 18W Microfit	*Section 3.6.7.1*	RS232 and RS485 Full Duplex Communications and Input Power Ethernet (100 Mbit/s) Communications and Input Trigger
C1	MOLEX 10W Microfit	*Section 3.6.7.2*	RS232 Communications, Input Trigger and Power Pass Through
C2	MOLEX 10W Microfit	N/A	Sonardyne internal use only
T1	MOLEX 8W Microfit	*Section 3.6.7.3*	RS232 and RS485 Half Duplex Communications, Output Trigger and Power Pass Through
T2	MOLEX 8W Microfit	*Section 3.6.7.3*	RS232 and RS485 Half Duplex Communications, Output Trigger and Power Pass Through
Battery	MOLEX 2W Microfit	N/A	Used for optional, Sonardyne provided battery pack

3.6.6.2 Power

Input: 24/48 V dc, 15 W nominal, (35 W max with optional external battery).

3.6.6.3 Shock Vibration and Temperature

Operating Temperature: -20 to +55°C

Storage Temperature: -20 to +60°C

Shock Rating (Operational): 22 g, 11 ms half sine.

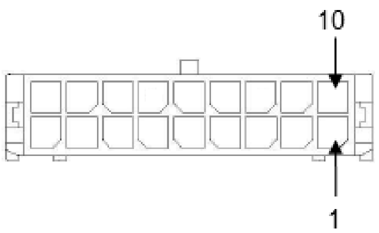
3.6.6.4 Weight

In Air: 7.8 kg (Estimated)

3.6.7 Detailed Pinouts

3.6.7.1 SPRINT CP/E1 Port

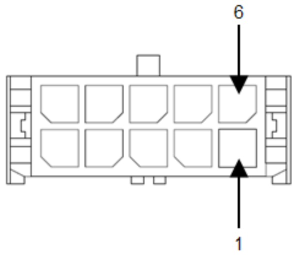
Face View (PCB Socket)



MOLEX Pin No.	Function
1	TX TX-
2	RX TX+
3	RX-
4	Ethernet RD -
5	Ethernet TD -
6	RX+
7	Trigger In
8	Comms / Trig Ground
9	N/C
10	DC In
11	DC In
12	DC 0 V
13	DC 0 V
14	Ethernet RD +
15	Ethernet TD +
16	RS232 / 485 Select **Connect to 0 V / Pin 8 for RS232** **Do not connect for RS485**
17	Comms / Trig Ground
18	N/C

3.6.7.2 SPRINT C1 Port

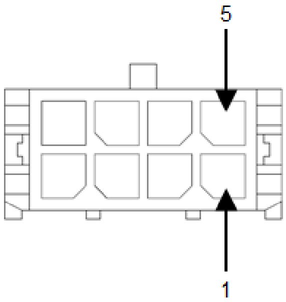
Face View (PCB Socket)



MOLEX Pin No.	Function
1	DC Out
2	RX
3	N/C
4	Comms / Trig Ground
5	N/C
6	DC 0 V
7	TX
8	N/C
9	Trigger In
10	N/C

3.6.7.3 SPRINT Transceiver (T1 / T2) Port

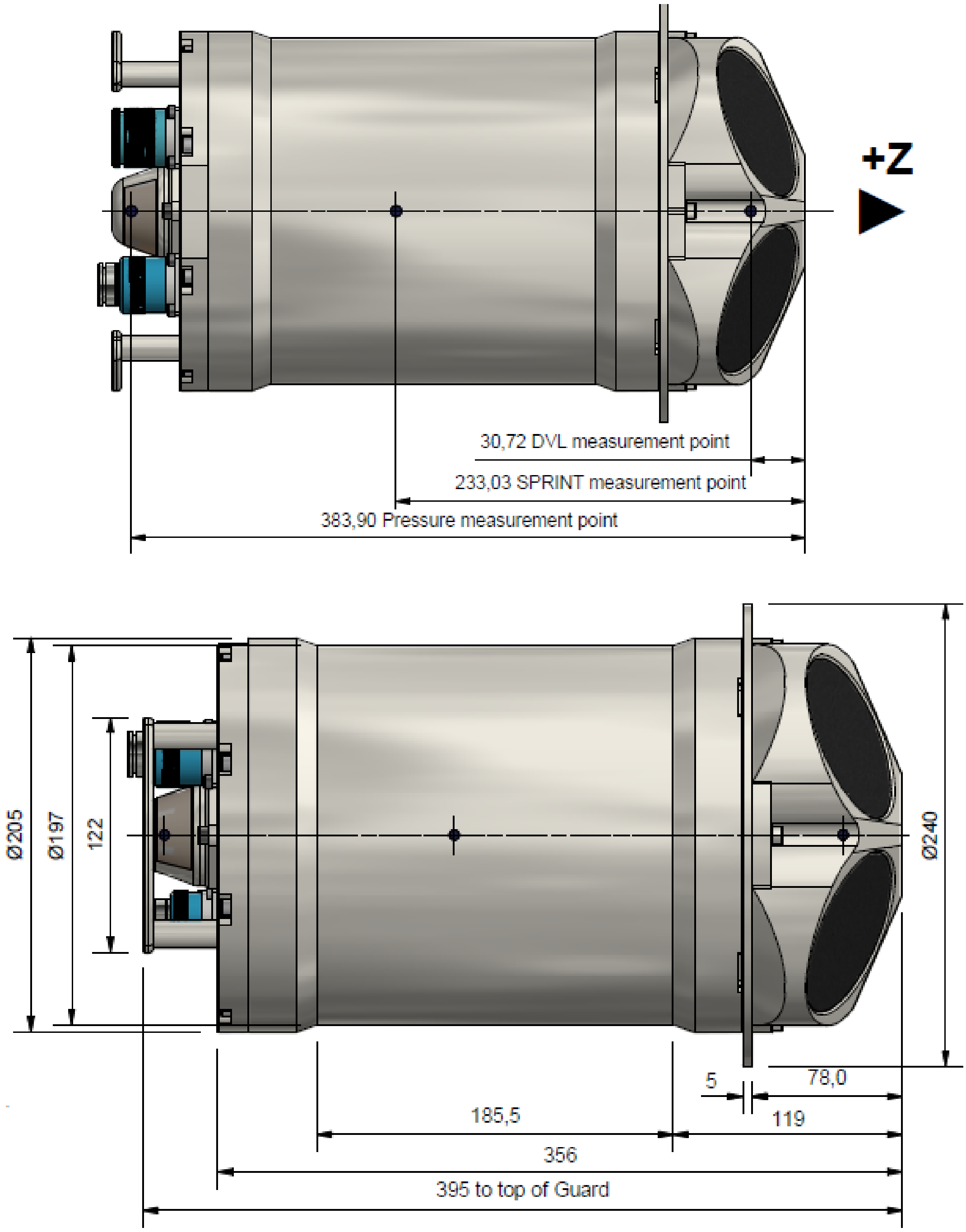
Face View (PCB Socket)

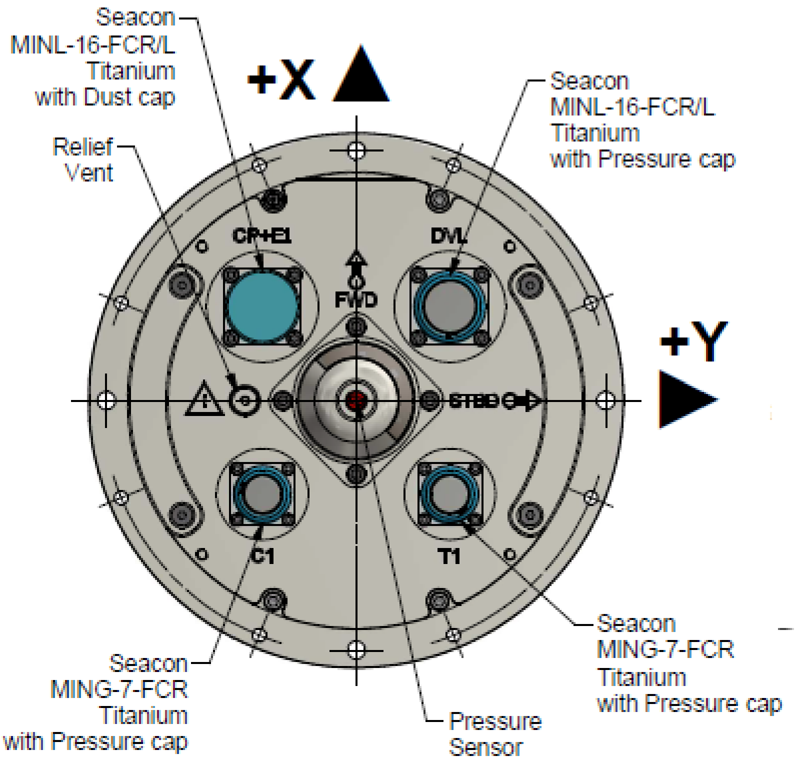


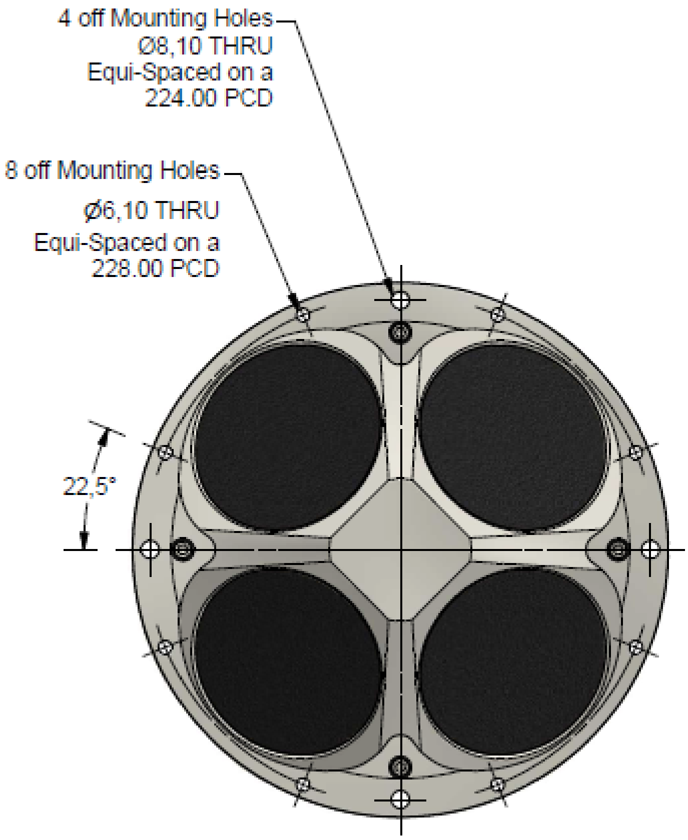
MOLEX Pin No.	Function
1	DC Out
2	RX TX+
3	Trigger Out
4	N/C
5	DC 0 V
6	TX TX-
7	Comms / Trig Ground
8	N/C

3.6.8 SPRINT-Nav 300, 500 & 700 4K

Figure 3-7 SPRINT-Nav 300/500/700 4K Outline Drawing




Seacon
MINL-16-FCR/L
Titanium
with Dust cap
Relief
Vent
+X
Seacon
MINL-16-FCR/L
Titanium
with Pressure cap
CP+E1
DVL
FWD
+Y
STBD
C1
T1
Seacon
MING-7-FCR
Titanium
with Pressure cap
Seacon
MING-7-FCR
Titanium
with Pressure cap
Pressure
Sensor


4 off Mounting Holes
Ø8,10 THRU
Equi-Spaced on a
224.00 PCD
8 off Mounting Holes
Ø6,10 THRU
Equi-Spaced on a
228.00 PCD
22,5°

3.6.8.1 Connector Pinouts

Table 3-3 SPRINT-Nav 300, 500 & 700 4K Connector Functions

Port	Connectors			
	Type	**Internal Sensor**	**Detailed Pinout (Section)**	**Function**
CP/E1	Seacon MINL-16FCR/L	SPRINT INS	*Section 3.6.10.1*	RS232 and RS485 Full Duplex Communications and Input Power Ethernet (100 Mbit/s) Communications and Input Trigger
C1	Seacon MING-7-FCR	SPRINT INS	*Section 3.6.10.2*	RS232 Communications, Input Trigger and Power Pass Through
T1	Seacon MING-7-FCR	SPRINT INS	*Section 3.6.10.3*	RS232 and RS485 Half Duplex Communications, Output Trigger and Power Pass Through
DVL	Seacon MINL-16FCR/L	Syrinx DVL	*Section 3.6.10.4*	Ethernet, serial, Input Trigger and Alternative Power (see Note)

Note

The DVL connection will not normally be required for operation as DVL communications and power will be routed internally via the SPRINT. It allows an independent connection and power to the DVL if required. If the DVL is to be powered via the DVL connector the user should ensure that the power routed via the SPRINT is not enabled/connected.

3.6.8.2 Power

Input: 24 V dc, 27 W nominal, 63 W max (excluding external sensors).

3.6.8.3 Shock Vibration and Temperature

Operating Temperature: -20 to +55°C

Storage Temperature: -20 to +55°C

Shock Rating (Operational): 22 g, 11 ms half sine.

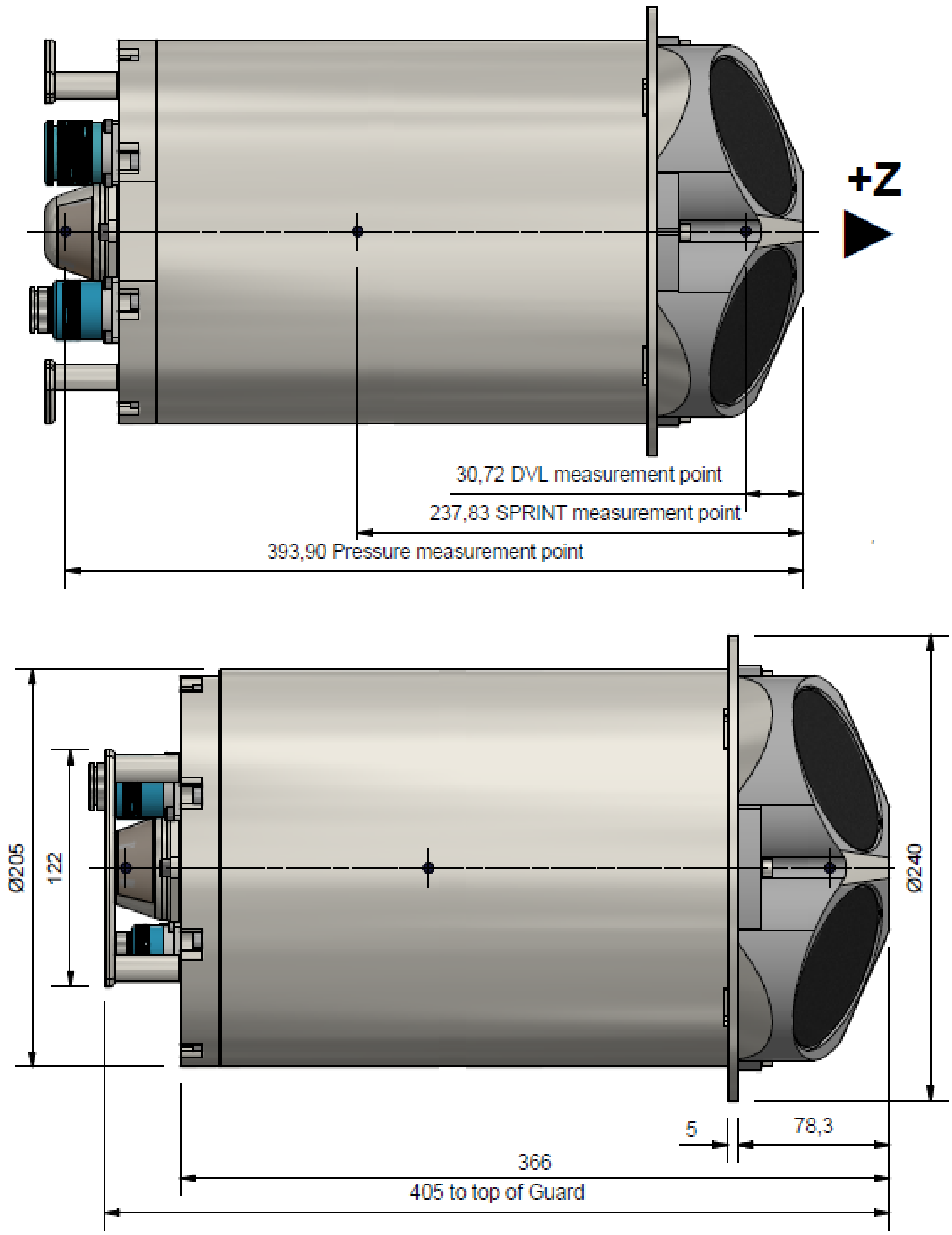
3.6.8.4 Weight

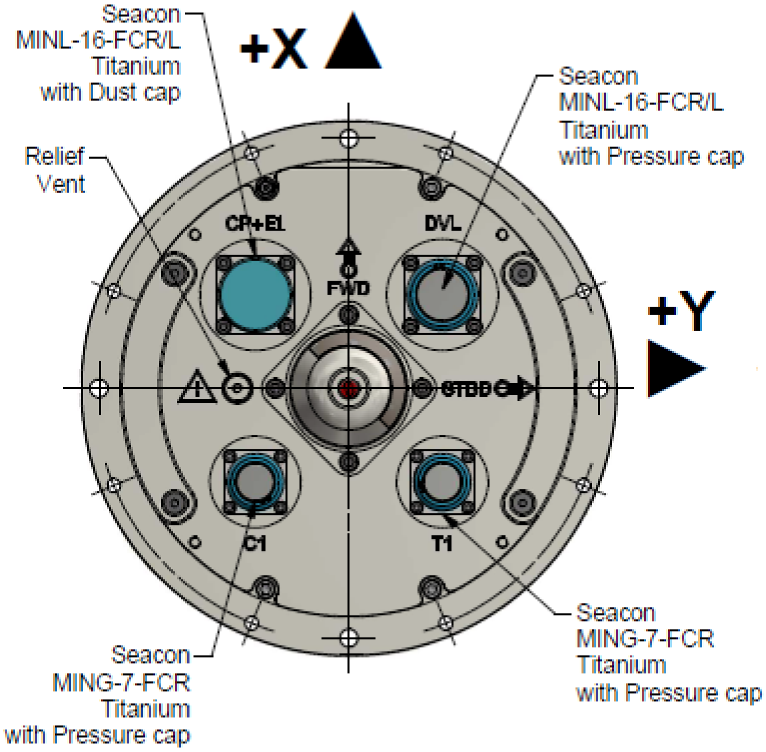
In Air: 23.9 kg

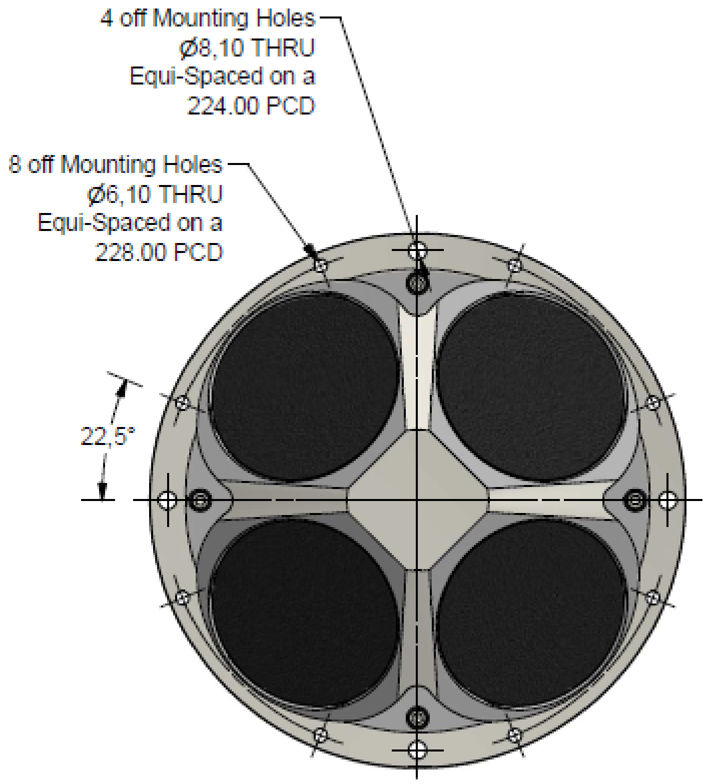
In Water: 13.1 kg

3.6.9 SPRINT-Nav 300, 500, & 700 6K

Figure 3-8 SPRINT-Nav 300/500/700 6K Outline Drawing




Seacon
MINL-16-FCR/L
Titanium
with Dust cap
+X
Relief
Vent
Seacon
MINL-16-FCR/L
Titanium
with Pressure cap
CP+E1
DVL
FWD
+Y
STBD
C1
T1
Seacon
MING-7-FCR
Titanium
with Pressure cap
Seacon
MING-7-FCR
Titanium
with Pressure cap


4 off Mounting Holes
Ø8,10 THRU
Equi-Spaced on a
224.00 PCD
8 off Mounting Holes
Ø6,10 THRU
Equi-Spaced on a
228.00 PCD
22,5°

3.6.9.1 Connector Pinouts

The connector pinouts are identical to the SPRINT-Nav 300, 500 & 700 4K; see *Section 3.6.8.1*.

3.6.9.2 Power

The power requirements are identical to the SPRINT-Nav 300, 500 & 700 4K; see *Section 3.6.8.2.*

3.6.9.3 Shock Vibration and Temperature

The environmental specifications are identical to the SPRINT-Nav 300, 500 & 700 4K; see *Section 3.6.8.3.*

3.6.9.4 Weight

In Air: 28.1 kg

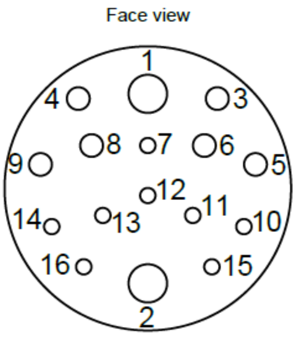
In Water: 17.2 kg

3.6.10 Detailed Pin Outs

Detailed connector pin outs are described in the following sections.

3.6.10.1 SPRINT CP/E1 Port (16-Pin Seacon)

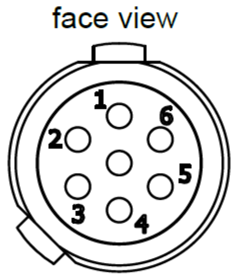
Table 3-4 SPRINT CP/E1 Port (16-Pin Seacon) Pinout



Seacon Pin No.	Function
1	DC 0 V
2	DC In
3	Comms/Trig Ground
4	Screen
6	Trigger In
7	Ethernet TD -
8	RS232/485 Select **Connect to 0 V1/Pin3 for RS232** **Do not connect for RS485**
10	Ethernet RD -
11	RX TX+
12	Ethernet TD +
13	TX TX -
14	RX -
15	Ethernet RD +
16	RX +

3.6.10.2 SPRINT C1 Port (7-Pin Seacon)

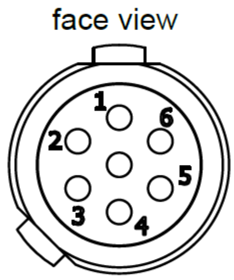
Table 3-5 SPRINT C1 Port (7-Pin Seacon) Pinout



Seacon Pin No.	Function
2	Comms/Trig Ground
3	Trigger In
4	DC 0 V
5	TX TX-
6	RX TX+
7	DC Out

3.6.10.3 SPRINT Transceiver (T1/T2) Port (7-Pin Seacon)

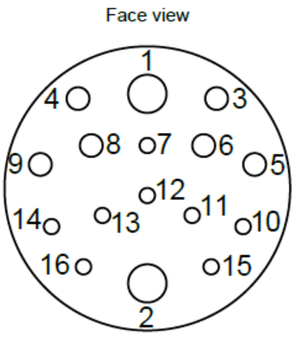
Table 3-6 SPRINT Transceiver (T1/T2) Port (7-Pin Seacon) Pinout



Seacon Pin No.	Function
2	Comms/Trig Ground
3	Trigger Out
4	DC 0 V
5	TX TX-
6	RX TX+
7	DC Out

3.6.10.4 Syrinx DVL Port (16-Pin Seacon)

Table 3-7 Syrinx DVL Port (16-Pin Seacon) Pinout



Seacon Pin No.	Function
1	DC 0 V
2	DC In
3	Comms/Trig Ground
4	Screen
6	Trigger In
7	Ethernet TD -
8	No Connection
10	Ethernet RD -
11	RS232 RX
12	Ethernet TD +
13	RS232 TX
14	No Connection
15	Ethernet RD +
16	No Connection

Section 4 – Lodestar/SPRINT Concepts and Definitions

4.1 Time

4.1.1 Instrument Time (System Time)

Lodestar/SPRINT contains a high precision clock oscillator to produce a 1 µs counter. Internally this is represented by a 64-bit unsigned integer, but it is usually represented by a 48-bit unsigned integer (corresponding to a maximum of approximately 3258 days) in any output. This counter starts at 0 after every reset of the instrument.

Note

In past Lodestar/SPRINT documentation the term "System Time" was used, this is the equivalent to "Instrument Time".

4.1.2 Common Time (UTC)

Lodestar/SPRINT contains a battery backed up Real Time Calendar (RTC). This is set up at the factory with Coordinated Universal Time (UTC) time and day information. This stored value of UTC can also be updated by messages and commands to the instrument.

4.2 Frames

All reference frames are right hand and orthogonal. See *Section 4.3.2* for details as to the calculation required to move from one frame to another.

4.2.1 IMU Body Frame

X = Forward

Y = Starboard (Right)

Z = Down

4.2.2 Vehicle

X = Forward

Y = Starboard (Right)

Z = Down

4.2.3 Navigation (NED)

X = North

Y = East

Z = Down

4.3 Lever Arms and Mounting Angles

4.3.1 Lever Arms

Each Instrument or sensor on a vehicle or vessel has a position offset and possible misalignment in the vehicle coordinate frame. The position offset is an X, Y, Z distance from the Central Reference Point (CRP). This is also known as a lever arm. The Lodestar/SPRINT in general will also have a lever arm and misalignment to the vehicle frame. Standard practice is to set them as zero to use the Lodestar/SPRINT as the reference point of the vehicle. The lever arms are distances from the CRP in the forward, starboard and down directions of the vehicle or vessel and are measured in metres.

Note

When modifying Lever Arms for an instrument or sensor, one or more algorithms may require a reset. Where this is required it is noted in the section documenting the configuration of the instrument or sensor.

4.3.2 Mounting Angles

Sensor mounting angles are entered using the “<sensor> MA alpha beta gamma” command (e.g. DVL MA) and express the Euler rotation angle sequence from vehicle frame into sensor frame.

Note

When modifying Mounting Angles for an instrument or sensor, one or more algorithms may require a reset. Where this is required it is noted in the section documenting the configuration of the instrument or sensor.

4.3.2.1 Euler Angles (Tait Bryan Rotations)

The “Tait Bryan” rotations given hereafter are commonly and henceforward referred to as the ***Euler angles***.

The Euler angle rotation sequence from NED to IMU Body frame is:

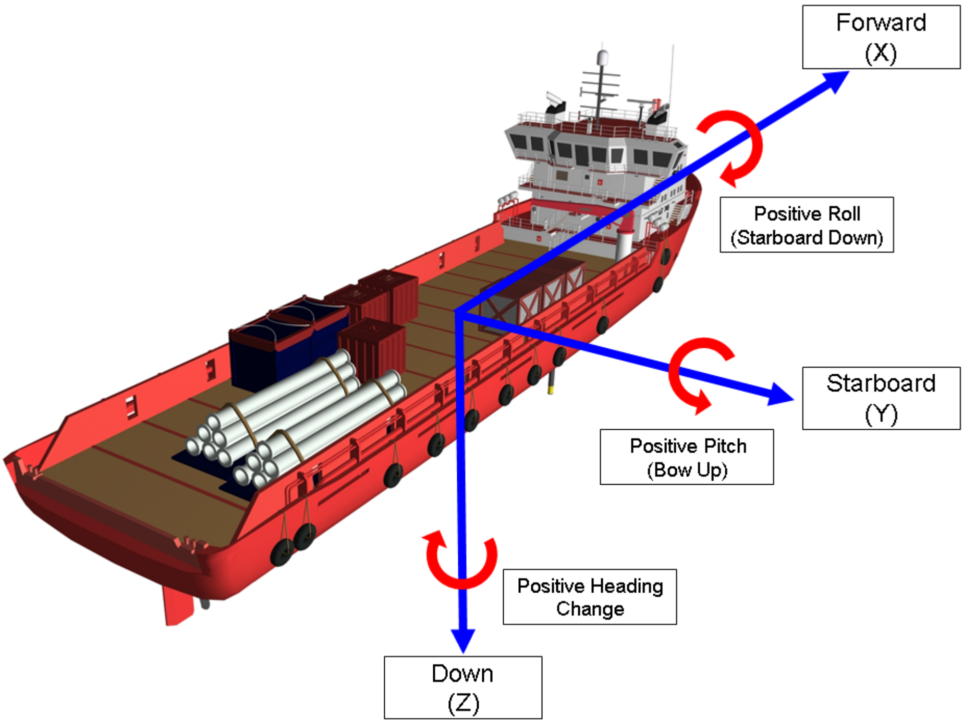
1. Rotation by the ***heading*** angle ***ψ*** (psi) about Zned.
2. Rotation by the ***pitch*** angle ***θ*** (theta) about the resulting Y axis.
3. Rotation by the ***roll*** angle ***φ*** (phi) about the resulting X axis.

Similarly, the rotation sequence from a reference frame (Vehicle) to a sensor frame (IMU, USBL, DVL) is:

1. Rotation by the ***gamma*** angle about Zref.
2. Rotation by the ***beta*** angle about the resulting Y axis.
3. Rotation by the ***alpha*** angle about the resulting X axis.

4.4 Coordinate System

Figure 4-1 Lodestar/SPRINT Coordinate Frame



4.5 Navigation Algorithms

4.5.1 Inertial Measurement Unit (IMU)

The core Inertial Sensor Assembly (ISA) block is composed of orthogonal triads of Ring Laser Gyroscopes (RLGs) and pendulous force rebalanced accelerometers. The gyroscopes measure angular rate and the accelerometers measure linear acceleration (specific force). These specific types of sensors are widely used in safety critical applications such as civil aviation and were chosen for their extremely high quality and unrivalled field proven Mean Time Between Failure (MTBF). Performance is limited by export regulations, not the sensor technology; sensors with higher performance and identical form, fit and function are available and could be used where operation under ITAR is acceptable. The sensor error characteristics are highly stable and the sensors are insensitive to temperature variation and mechanical vibration often experienced in subsea robotic applications and other challenging tasks.

Front-end sampling of the inertial sensors is done at a rate in excess of 1 kHz. The Inertial Measurement Unit (IMU) functionality compensates the raw measurements via a host of factory calibration parameters including temperature, misalignment, non-linearity, bias, scale factor, accelerometer size effect and also applies a loss-less reduction of frequency (“Conning & Sculling”) resulting in fully compensated “delta angles” and “delta velocities” at a default rate of 100Hz.

The IMU data drives two complementary and largely independent navigation algorithms:

- Lodestar Attitude and Heading Reference System (AHRS)
- SPRINT Inertial Navigation System (INS)

The AHRS is almost entirely self-contained and robustly computes attitude, heading and heave. The INS supports optimal integration of measurements from a wide range of external aiding sensors to output highly accurate vehicle position, orientation, velocity, angular rate and linear acceleration.

AHRS orientation is nominally correct from about 5 minutes after power up and remains available throughout operation. Orientation from the AHRS is used to seed the INS thereby bypassing what in other systems may be referred to as the “INS coarse alignment” phase. Since the AHRS is always available, INS heading alignment is practically instantaneous should there ever be a need to reset the INS. Furthermore, there are no restrictions on dynamics during initialisation of either the AHRS or the INS.

4.5.2 Attitude and Heading Reference System (AHRS)

The AHRS provides the combined functionality of a marine gyrocompass (Heading) and a Motion Reference Unit (MRU: roll, pitch and heave). The AHRS works almost entirely autonomously by sensing the direction of gravity (=> down) and how it changes relative to inertial space as the Earth rotates (=> East). Accuracy is improved by providing coarse Latitude and velocity to the AHRS.

Latitude can be entered and stored via the Command & Control (C&C) interface or updated automatically by feeding in suitable position aiding data. A latitude error of 5km has negligible effect on the AHRS output. Velocity aiding is possible by supplying a suitable $VTG telegram to the Lodestar/SPRINT.

Note

Operation without velocity compensation and up to 100 km of latitude error is acceptable when the AHRS is only used to seed the INS, as is typically the case for AUV and ROV use.

4.5.3 Inertial Navigation System (INS)

INS is provided in Sonardyne’s SPRINT product line up – there is a usually an upgrade option to turn a Lodestar into a SPRINT. The Sonardyne regional offices can provide guidance on these options.

The Inertial Navigation System (INS) “integrates” IMU data to calculate velocity, orientation and position. Inertial navigation by itself would degrade over time: An error state Kalman filter estimates the errors of the INS by statistically optimal integration of measurements from external aiding sensors. The determined errors are continuously applied to the INS as corrections in a closed loop Aided INS configuration. Given that most aiding supplied to a SPRINT is acoustic in nature the acronym AAINS (Acoustically Aided INS) is sometimes used. The Kalman filter also estimates and compensates certain slowly time varying errors in the inertial and aiding sensors. This improves accuracy and integrity beyond what would be possible via factory calibration only.

Adding an acoustic Doppler Velocity Log (DVL) to the SPRINT INS provides a very practical autonomous dead-reckoning navigation capability by limiting INS drift rate to just a few meters per hour. The INS can also reduce the noise of Ultra-Short Baseline (USBL) positioning and improve the operational efficiency of Long Baseline (LBL) positioning by supporting the use of sparse seabed transponder arrays. In the vertical, INS will reduce dynamic effects on pressure depth such as wave motion and transient vehicle hydrodynamic effects.

4.6 Aiding Sources

Table 4-1 gives an overview of the types of aiding that will be accepted by the INS.

Table 4-1 INS Aiding Sources

Aiding Source	Aiding Type			Aiding Considerations	
	Position Aiding (Horizontal)	Velocity Aiding	Depth Aiding (Vertical)	Time Sync Required?	Can be used to Initialise INS?
USBL (SUSBL)	✓	×	Optional	Optional	✓
XPOS (External Position)	✓	×	Optional	Optional	✓
GNSS (GPS)	✓	×	Optional	✓	✓
LBL	✓	×	×	✓	×
DVL	×	✓	×	×	×
Zero Velocity	×	✓	×	×	×
Pressure Depth	×	×	✓	×	✓
Zero Mean Depth (ZMD)	×	×	✓	×	✓

4.6.1 External Position Aiding (XPOS)

The INS can be position aided (including initialisation) with a generic external position update. This may be a manual position fix or a way-point from the AUV/ROV control.

4.6.2 USBL Aiding (SUSBL)

A remote vessel-mounted USBL transceiver can measure the range and bearing to an acoustic transponder fitted to a subsea vehicle. Combined with the vessel's GNSS (GPS) the position and orientation sensors provide the absolute position of the subsea vehicle. This position can be transmitted to the subsea vehicle INS via acoustic telemetry or a tether. It is important to ensure that the USBL and INS have access to and make use of a common time base such as UTC. Whilst the use of Sonardyne USBL provides a tighter acoustic-inertial integration and the best possible USBL positioning performance, SPRINT can accept position aiding from any USBL system that uses an industry standard telegram. Although SPRINT improves USBL system precision and short term accuracy, it will not resolve any inherent systematic errors that are present. Users must therefore make sure the USBL system they are using is correctly calibrated and that recommended operating practices are observed, for example, using regular sound velocity profiles.

Note

"SUSBL" is used as the aiding sensor name for the type of aiding above. The command and control also has "USBL" configuration options; this is for when the Lodestar/SPRINT is used with (local) USBL aiding as would be the case on a DP-INS vessel.

4.6.3 GNSS Aiding (GPS)

A vehicle mounted GNSS receiver can be used to provide position aiding to the INS when the vehicle is at the surface. This can be particularly useful when logged data from a mission is post processed: A high accuracy GNSS position fix at the end of an operation can be used to smooth the stored data "backwards in time". It compensates for the drift that occurred earlier.

4.6.4 LBL

In Sparse LBL operations, two or more (instead of four or five) beacons are deployed on the seabed and their positions derived using ‘box-in’ or other top-down calibration techniques. With a known beacon position, the INS can navigate in Sparse LBL mode using the ranges from one or more seabed deployed beacon to acoustically aid the INS and constrain error growth in the absolute position output.

4.6.5 Zero Velocity Update Aiding (ZUPT)

In certain operational situations the subsea vehicle may be static (e.g. during average position fixes). In these situations, particularly if there is a risk of loss of other aiding, it may be beneficial to aid the INS with ‘zero velocity’. This bounds the velocity error and so reduces rate of position drift.

4.6.6 Zero Mean Depth (ZMD)

This depth aiding mode assumes that the Lodestar/SPRINT is at a constant average depth, for example when a vehicle is at the surface.

4.6.7 Vehicle-Mounted Sensor Aiding

Lodestar/SPRINT have the ability to use vehicle-mounted aiding sensors such as Doppler velocity logs (DVL) and pressure/depth sensors. The use of these sensors provides further benefits for subsea navigation such as the ability to provide precise and continuous navigation output even if external acoustic positioning is lost for periods of time. Lodestar/SPRINT does not need to be physically co-located with the Doppler Velocity Log or integrated into the same housing. Only ‘coarse’ alignments and offsets from the INS to the DVL are needed from the user. Fine offsets and misalignments are then calculated in the field using a calibration routine. This approach allows for more flexible mounting configurations to be considered.

Section 5 – Protocol and Messaging Overview

5.1 Multiplex Format

Lodestar supports communication of ASCII and binary messages via multiplexed serial and Ethernet links. Each message is given a unique ID and provided as the payload data within a packet, i.e. one packet contains one message. Separation of packets is achieved via unique packet start and end byte sequences and the “byte stuffing” technique (see *Section 5.1.1*). This is best used when one or more binary messages are to be transmitted, as the multiplex protocol provides the unique byte sequence to allow software parsers to correctly interpret the transmitted data stream. It is also possible to configure a port to be non-multiplex, in this case only log messages are modified with a wrapper (see *Section 5.2.3*) with all other message payloads being transmitted without modification.

5.1.1 Byte Stuffing

The scheme used for separation of packets is known as *byte stuffing*. The start and end of a packet is marked with unique ASCII control bytes sequences, namely Data Link Escape (DLE) (0x10) & STX (0x02) at the start, and DLE (0x10) & End of Text (ETX) (0x03) at the end. For every DLE byte found *within* the packet data an extra DLE byte is inserted (stuffed). This allows a parser of the resulting data stream to distinguish between bytes within a packet having the value DLE (0x10) and the start/end of a packet.

Table 5-1 Packet Start and End Characters

Name	Byte Value	Description
DLE	0x10	Data Link Escape
STX	0x02	Start of Text
ETX	0x03	End of Text

Example byte stuffed data (e.g. transmitted via serial link):

```
0x65, DLE(0x10), DLE(0x10), ETX(0x03)
```

Here the DLE/ETX pair does NOT indicate the end of a packet but rather the (original) byte sequence:

```
0x65, 0x10, 0x03
```

Further examples can be seen in *Section 7.3 and Section 7.5*

5.1.2 Packet Format

Table 5-2 Multiplex Packet Format

DLE STX	ID	Port Pass/Snoop Info (optional)	Timestamp (optional)	Payload	Checksum	DLE ETX
2 Bytes	2 Bytes	2 or 4 Bytes	6 Bytes	0-2047 Bytes	1 Byte	2 Bytes

The following sections provide details of the different fields of the packet format shown in *Table 5-2*.

5.1.2.1 DLE STX

This denotes start of packet.

5.1.2.2 ID

The following table indicates the sub-fields within the 2 byte ID field.

Table 5-3 Multiplex ID Field Format

ID			
TS	RES	SID [0-15]	MID [0-1023]
1 bit	1 bit	4 bits	10 bits (LSB)
2 Bytes			

- TS – If set (‘1’) indicates that the Timestamp field is present and prepended to the payload.
- RES – Reserved, any value to be expected
- SID – Depending on message type this is “SenderID”, “SourceID” or “SystemID” (see *Section 5.1.2.3*).
- MID – Message IDentifier, number that indicates which type of message is contained within the payload

5.1.2.3 SID

The SID field depends on the message type and direction (from or to the Lodestar/SPRINT instrument) of the multiplex packet.

- Port Snoop data – the SID will represent the port (0–4 for UART, 15 for Ethernet) that the data has been snooped from.
- Port Pass data – the SID will represent the port (0–4 for UART, 15 for Ethernet) that the data is being received on: see *Section 7.11.5.2* for an example
- Output Message – the SID will represent the Remote Point configured for the output; see *Section 7.8.2* for an example.
- Input & Log Message – the SID is generally 0 but where necessary it is used by splitting the field in two, with the two MSBs of the SID representing the source sensor type (for example GPS GGA or SUSBL GGA; see *Sections 7.5.2.1 & 7.5.2.2*) and the two LSBs representing sensor number (usually 0, but for LSZDA output would be set to 1; see *Section 7.11.6*).

5.1.2.4 Port Pass/Snoop Info

This field is conditional on the MID in the ID field.

If the MID indicates that the payload contains Port Pass data there will be four bytes appended after the ID field, the first two bytes are the source port number (least significant byte first) with the final two bytes containing the destination port number (least significant byte first).

If the MID indicates that the payload contains Port Snoop data there will be two bytes appended after the ID field, these two bytes indicate the port (least significant byte first) the snoop data has been captured from.

5.1.2.5 Timestamp

Packet timestamp using the Instrument Time with the least significant bit equalling 1 µS. Actual definition is message dependent and cannot be assumed to be time of validity of the message payload. Only for output/logging and should not be present on input messages.

5.1.2.6 Payload

Message data.

5.1.2.7 Checksum

This is the byte-wise exclusive-OR of the pre-byte stuffed fields: ID, Port pass/snoop info, Timestamp and Payload.

5.1.2.8 DLE ETX

This denotes end of packet.

5.1.3 Code Example

A C code extract is shown below that removes the multiplex protocol, indicates when a complete message has been received and extracts the message ID. Note that this does not include checksum calculation or error handling. The code assumes that the received data is read into variable ‘`Databyte`’ one byte at a time.

```
switch (Databyte){
    case 0x10:  //DLE
        if (DLEReceived == false){
            //first DLE  wait for the next DLE
            DLEReceived = true;
        }else{
            //DLE DLE found  copy one DLE to message buffer
            DLEReceived = false;
            message[datalength++] = Databyte;
        }
        break;

    case 0x02: //STX
        if (DLEReceived == true){
            // DLE STX  Start of the message found
            DLEReceived = false;

            DLE_STX_Found = true;
        }else{
            // Databyte is data
            message[datalength++] = Databyte;
        }
        break;

    case 0x03: //ETX
        if (DLEReceived == true){
            //DLE ETX found
            DLEReceived = false;
            if (DLE_STX_Found == true){
                  //DLE-STX/DLE-ETX pair found - complete message
                  MessageComplete = true;
            }
            DLE_STX_Found = false;
```

```
        }else{
            //Databyte is data
            message[datalength++] = Databyte;
        }
        break;

    default:    //Data
        if (DLEReceived == true){
            DLEReceived = false;
        }
        message[datalength++] = Databyte;
        break;
}

//Extract ID and checksum from complete message
if (MessageComplete == true)
{
    //get Source Id xx(2bits Source ID) (2bits Sensor ID)xx
    sourceId = (message[0] & 0x00) >> 4;

    //get Sensor Id
    sensorId = (message[0] & 0x3C) >> 2;

    //get Message ID
    mid = ((message[0] & 0x03) << 8) | (message[1]));

    //get chksum
    Chksum = message[datalength  1];
}
```

5.2 Inputs, Outputs and Logs

“Input” messages cover all inputs to the instrument that the instrument should be using; “Commands” can be considered a special type of “Input” message. Both “Output” and “Log” messages are outputs from the instrument, the difference being that “Outputs” are output at a specified user defined rate whilst “Logs” are event driven messages.

5.2.1 Typical Input Processing

The typical processing consists of the following; a message must “pass” each stage to allow it to progress on to the next.

1. Message format is checked for consistency against the message specification, and checksum or CRC is checked.
2. “Pre-filter” checks – the data within the message is checked against the current configuration to check that all data is within the limits set.
3. Message data is supplied to one or more algorithms where further checks could be made before it is used as aiding data.

5.2.2 Output Messages

As well as there being many choices of output messages that are supported there are a number of other configurable options that are detailed below.

5.2.2.1 Output Rate

The desired output rate must be specified (in Hz) at the time an “Output” message is configured.

5.2.2.2 Remote points

A number of remote points can be configured by specifying the lever arm from the vehicle CRP to the remote point.

This could be used where the integrator would like the data in a message to relate to a position on the vehicle that is not the CRP, for example the location of a payload instrument. With a remote point configured this can be specified when configuring an “Output” message such that the output will be with respect to that point on the vehicle.

Note

Some data parameters within a message may not be converted for a remote point – where this is the case this is identified in the message specification.

5.2.2.3 Algorithm Source

The algorithm from which to source the message data is also configurable, for some messages the source is fixed as only being available from AHRS or INS, but some can take data from either algorithm.

Where available, it is recommended that heading, pitch and roll information is obtained from the INS algorithm since this is generally more accurate than the AHRS. The AHRS will initialise the INS and is subsequently used as an automatic integrity check via simple comparison. Since the AHRS operates largely off the self-contained IMU data, it is generally more reliable than the INS.

5.2.3 Log Messages

When “Log” messages are output on a non-multiplex port, most messages are output with an ASCII wrapper. This wrapper adds a timestamp field indicating the time corresponding to the logged message and a checksum. The format of the wrapper is shown below with the field definitions in *Table 5-4*.

:hhhhhhhhhhhh,{xxxxxx}*hh<cr><lf>

Table 5-4 LOG message ASCII wrapper format

Field	Description
:	Start Character
hhhhhhhhhhhh	6 byte timestamp, in hex-ASCII representation
,{	Delimiter and start of payload
xxxxxx	Message data
}	End of payload
*hh	Terminator and checksum
<cr><lf>	Carriage Return and Line Feed

Notes

Depending on the message type the timestamp is the time of arrival of the logged message or the time of validity if the message contents are not related to an input message.

Terminating <cr><lf> characters in the message data will not be replicated in the payload section (if the message data is ASCII).

Not all messages will have the wrapper applied for example a $ZDA telegram produced from the Lodestar/SPRINT will not have the wrapper applied. The message definitions should be consulted for messages that will not use the ASCII Log wrapper.

The checksum is an exclusive-OR of the payload including the start and end brace (‘{‘ & ‘}’) characters

Section 6 – Command and Control Overview

6.1 Port Commands

For commands to be accepted on a particular port, the port must be configured to accept commands as an aiding source. Two ports are configured to always support commands, these are:

- Port 0 (Console)
- Port 4000 (Ethernet)

To enable commands on another port, use the command:

```
IN <port> MSG + COMMAND
```

When connected to a port that is configured to accept commands.

Although there can be multiple ports configured to accept commands only one command can be actioned at a time across all ports. It is therefore important to ensure that a response to indicate a command has been actioned has been received before issuing another command.

The command syntax is not case sensitive, however all commands in this document will be in uppercase (unless otherwise stated), if the responses are listed these will be as they would be sent by the Lodestar/SPRINT.

6.2 Non-Multiplex

On a non-multiplex port command mode can be entered (started) by entering the following string:

```
<Ctrl-o>SON
```

(hold down the ‘Ctrl’ key and press ‘o’, then release the keys and type SON)

To exit, the ’escape’ key should be pressed:

```
<ESC>
```

When in command mode, if an output message is configured on that port it will be suspended until command mode is exited, however if a Log message is configured it will continue to be output on the port.

Some commands require command mode to be exited before their corresponding action is performed, for example, changing the baud rate on a port that is currently being used to enter commands.

6.3 Multiplex

On a multiplex port there is no need to “enter” or “exit” command mode. Commands can be sent as a multiplex packet with MID 0 to indicate that the payload contains a command; see Section 7.7 for examples).

6.4 Command Types

Commands are used to interrogate or to set configuration parameters within the Lodestar/SPRINT. When interrogating a configuration parameter or group of parameters the response is typically returned immediately. This is less true for commands that change certain configuration parameters where there may be a small delay in completing the request. The completion of a command is indicated by the Lodestar/SPRINT returning one of the following responses:

```
ok
```

```
not ok
```

In either case the response to a command may contain multiple other lines of text that precede one of the command terminators above.

Section 9 – Command and Control gives an overview of the different commands that are available and their syntax, but in general the following syntax holds true (delimitation between fields is by whitespace this is indicated by `<SPACE>)` :

```
<CommandGroup><SPACE><Command><SPACE><Param1>…<ParamX><cr><lf>
```

Where `<CommandGroup>` could be one of the algorithms (“INS” or “AHRS”), aiding sources (“PRESS”, “SUBSL”, “GPS”, etc.) or the hardware platform (“SYS”, “LOG”, “OP”, etc.). Following this is the `<Command>` which is the name of the command and following are the parameter values that are required by the command (if any are required).

Note

Commands that affect the internal file system used to store log files will not be available until after such time that the file system initialises. This takes a short amount of time (approx. 30 s) after the firmware starts running.

6.4.1 Status Commands

Nearly all commands will act as status commands. If a command is also able to change configuration parameters it will act as a status command if no parameters are provided. The following example shows a purely status command and the response:

```
>>SYS UART

<<U0 Rx 0.3% Tx 0.3%

<<U1 Rx 0.0% Tx 0.0%

<<U2 Rx 0.0% Tx 0.0%

<<U3 Rx 0.0% Tx 0.0%

<<U4 Rx 0.0% Tx 0.0%

<<ok
```

The following example shows a configuration command being used to interrogate the current settings:

```
>>INS USE

<<INS USE DVL PRESS SUSBL

<<ok
```

6.4.2 Configuration Commands

When a command is being used to change the configuration, it is checked that it is formed correctly and that all mandatory parameter values are present and valid. When one or more parameter values are invalid, the whole command will be rejected (even if other parameter values are ok).

Whilst some commands are defined to alter a fixed number of parameters, some are used to work with lists. Examples of lists are “Outputs” and “Logs” defined on a port, or aiding sensors to be used by the INS. Where a command is modifying a list the “+” and “-“ characters can be used to add and remove items from the list, for example:

```
>>INS USE
```

```
<<INS USE GPS PRESS ZUPT
<<ok
>>INS USE - ZUPT
<<INS USE GPS PRESS
<<ok
```

When the addition or remove characters are not used, the command will behave as other commands, and assuming the parameter values are correct will use those going forwards, for example:

```
>>INS USE
<<INS USE GPS PRESS
<<ok
>>INS USE DVL
<<INS USE DVL
<<ok
```

Note

Where the configuration parameters are a list, the returned configuration may not match the entered list in terms of where items appear in the list, for example:

```
>>INS USE SUSBL PRESS DVL
```

Will return

```
<<INS USE DVL PRESS SUSBL
<<ok
```

6.4.3 Macros

6.4.3.1 Message Macros

Every message that can be used has a unique name within the command syntax. However, some messages are grouped together under a Macro name to simplify their use. *Table 6-1* defines the macros.

Table 6-1 Message Macros

Macro	Messages Included
GPS	GGA, VTG, ZDA, GLL, GNS, DTM, VBW, RMC, STN, GST, GSA
PSONLBLBCN	PSONBCNTIDE, PSONBCN, PSONLVR, PSONLOBS
SUSBL	PSIMSSB, GGA
DVL	PD4/5, PD0, ASONDV
SVS	SVS, PSONSS

It should be noted that if a macro name and an individual name are specified for logging on a port then two log messages are created, one for the macro and one for the message name, for example the following would produce two log `ZDA` messages per `ZDA` input message:

```
LOG 0 MSG GPS ZDA
```

6.4.3.2 Command Macros

Some commands will cause a number of other configuration parameters to be changed. An example of this behaviour is the `DVL TRIG` command, which when actioned would modify a number of `TRIG` parameters (as well as the `DVL TRIG` parameter).

6.5 Complete Lodestar/SPRINT Configuration

The complete configuration of a Lodestar/SPRINT can be obtained by sending the following command:

```
SYS LIST
```

This will return a complete list of all the configuration parameters of the unit. This is presented using the Command Syntax so the reply can be fed back in to the Lodestar/SPRINT as a series of commands (while adhering to the requirement to wait for a response from a command before sending a further command).

To save the current configuration the following command can be entered:

```
SYS SAVE FLASH
```

This saves the current settings to a “User” area of Non-volatile Memory. There are two “User” areas and the firmware keeps track of which one to use, if the most recently updated “User” area becomes corrupt it will fall back to use the other “User” area. Therefore for robustness two saves of the configuration should be completed such that both “User” areas have the same configuration. The current configuration is also saved automatically when a command is issued to shut down the Lodestar/SPRINT.

The following command can be used to restore the configuration from last save to the “User” area:

```
SYS LOAD FLASH
```

If a factory reset is required the following command will load a stored factory configuration, this is also the configuration that will be loaded if both “User” areas are found to not store valid configurations.

```
SYS LOAD FACTORY
```

Section 7 – Step-by-Step Integration (Example)

There are a multitude of possible combinations of connector and port usage with a Lodestar/SPRINT based instrument. The following guide provides an example of the steps required for integrating an OEM version of a Lodestar/SPRINT product. It may be the case that a particular integration only requires a subset of the functionality described below or that an integration will have to deviate substantially from that laid out below – in either case it is worth reading through the complete guide as it contains a number of hints for debugging an integration that whilst only mentioned once are applicable in multiple areas.

7.1 Initial Overview

Depending on the hardware there are likely to be numerous serial connections and Ethernet ports available for connection – inevitably some of these will be used for integration on the vehicle, however one or more may be free and unused. In the main, any port (serial or Ethernet) can be used for commands – however the CP and C1 ports are special in that they allow access to the Bootloader prompt. For this reason it would be beneficial to be able to use either CP or C1 initially as a control/debug port.

For the purposes of this example the following scenario will be true:

- The instrument in question will be a SPRINT 500 4K; see *Section 3.6.4*.
- CP will be connected to the vehicle control computer. Position, depth and time sync will be provided from the vehicle control computer.
- Ethernet will also be connected to a switch on the vehicle
- C1 is unused
- C2 is unused
- T1 is unused
- T2 is connected to a Syrinx DVL

For this scenario C1 is going to be used as the control/debug port. C1 will be configured to not use the Multiplex Protocol. The integrator can connect with their chosen terminal package e.g. Tera Term or similar) and can send commands and observe the Log and Output messages from the Lodestar/SPRINT. For information on Command Mode on a non-Multiplex port; see *Section 6.2*.

In the following sections unless explicitly required for an action to complete there will not be instructions to enter/leave command mode on the command/debug port. If there are instructions to enter commands, it assumed that, if not already in command mode, the debug port should be put into command mode. It is also advised that the configuration is regularly saved to the flash; see *Section 6.5*.

Note

Tera Term was used for examples in this manual. Some features of this package may not be present in other terminal packages; an example is the ability to log the raw serial communications to file.

7.1.1 Normal Boot Sequence

The normal boot sequence of a Lodestar/SPRINT immediately after power is applied is to execute the bootloader and wait for about 30s to allow a user to interrupt the boot sequence, before continuing to load the firmware and then executing the firmware once it is in memory. The firmware will start running automatically using the saved user configuration – if a user configuration is not available the firmware will revert to use a saved “Factory” configuration; see *Section 6.5*.

7.1.2 Resetting to Factory default

If required the instrument can be reset to use a Factory default configuration, if commands can be entered then the steps outlined in *Section 6.5* can be followed, the subsequent steps can be followed in other cases.

Note

If these settings are followed all stored user settings will be cleared and lost.

A serial connection to either CP or C1 is required. Assuming a terminal connection, the terminal should be configured for a baud rate of 9600. The next step requires the Lodestar/SPRINT to be turned on – but action will be required before the firmware loads.

With the power switched on to the Lodestar/SPRINT the following should be seen in the terminal:

```
Firmware Loader Version 1.04.00.35 Jul  3 2013 14:23:17

CPU UART FPGA 16387

CPU Interconnection FPGA 0:/00/10

SDHC Card capacity:-  8010072064 bytes

.........................
```

As soon as the series of dots start appearing on the screen the bootloader command mode can be entered (in the same way as the firmware on a non-multiplex port; see *Section 6.2*) by entering the following in the terminal:

```
<CTRL-o>SON
```

Following this the Bootloader command prompt should be seen, as shown below:

```
Lodestar:/>
```

To restore the factory settings enter the following command

```
LOADHEX CLRFLASH.HEX
```

The following specifies the output that should follow after the command above is entered:

```
CLRFLASH Version 1.00.00.02 Jan  4 2013 17:12:11

Erased FLASH area

PIC CONTROL 0 MATCH 1

PIC CONTROL 1 MATCH 1

PIC MATCH UNLK

Rebooting ...
```

The Bootloader should then start as normal and the normal boot sequence will be followed.

Note

An alternative to power cycling the Lodestar/SPRINT in starting the Bootloader is to send the following characters at 9600 Baud on either CP or C1 ports:

```
UNLK
```

This causes the Lodestar/SPRINT to reset.

7.2 Initial Configuration

Assuming the C1 is being used as the debug port and that factory defaults are loaded, a suitable terminal package should be used to connect to CP at 9600 baud and after entering command mode the following should be configured:

```
OP 1 MSG 0
LOG 1 MSG 0
OP 1 BAUD 9600
IN 1 MSG COMMAND
SYS SAVE FLASH
```

This will now allow the C1 port to be used as the debug/control port.

Note

***Section 7.11.1* provides detailed guidance detailing how to initally connect to a Lodestar/SPRINT where the configuration doesn't match the Factory configuration as it does at this point in the example integration.**

7.3 Multiplex AHRS Message Parsing

If the integration requires more than one message type to be passed on a port in the same direction, it is advised to implement the Multiplex Protocol for the transmission of those messages. Once a port is configured to use the Multiplex Protocol it is used for both receiving and sending messages on that port.

Once the firmware has booted without any user interaction the AHRS algorithm will start and even during the settling period will be outputting orientation data. The settling period of the AHRS algorithm can be interrogated and set by the following command

```
GC SETTLE
```

The default settling period is 600s, however if the initial integration is taking place in an office environment or where the Lodestar/SPRINT is not experiencing any movement this can safely be reduced to 180s.

Given that the AHRS algorithm is not dependent (see Note below) on any extensive configuration or external aiding it can be used as an output source to generate messages in the multiplex format. This can help an integrator to develop and test a Multiplex Protocol parser.

Note

The AHRS performance is dependent on having a correct Latitude provided, but even with an incorrect latitude, will still produce an output suitable for developing and testing a third party implementation of a Multiplex Protocol parser.

The following commands will configure the CP port to use the multiplex protocol and also set up a simple ASCII message sourcing data from the AHRS algorithm to be output at 1 Hz:

```
OP 0 MULTIPLEX 1
OP 0 BAUD 115200
OP 0 MSG SON2 1 SRC 0
LOG 0 MSG 0
```

To see example messages on the debug port similar to that being provided on the CP port, enter the following command:

```
OP 1 MSG SON2 1 SRC 0
```

On leaving command mode the SON2 message should appear at 1 Hz on the debug port, similar to the following:

```
:112143419 000133-000144 008272 040U
:112144418 000133-000144 008273 040U
:112145416 000134-000143 008274 040U
:112146414 000133-000144 008275 040U
```

The multiplex message that is being sent on CP is at least 51 bytes in length (assumes there has been no need to byte stuff the payload). The following table gives an indication as to how one multiplex message should be decoded.

Table 7-1 SON2 Multiplex Decode

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	80 78		Bit mask of 0x8000 to indicate if the time stamp is included Bit mask of 0x3FF to indicate the MID (0x3FF & 0x8078 = 0x78 = 120)
4–9	3E 1C A3 15 00 00		Timestamp in Instrument Time, the Byte 0x3E is the LS Byte.
10	3A	:	Start character of SON2 message
11–45	31 31 32 31 30 35 34 32 30 20 30 30 30 31 33 36 2D 30 30 30 31 34 34 20 30 30 38 32 35 38 20 34 32 55	112105420 000136-000144 008258 042U	Rest of SON2 message
46–47	0D 0A	<cr><lf>	End of SON2 Message
48	0D		Checksum
49	10		DLE (Data Link Escape)
50	03		ETX (End of Text)

Note

The example shown in Table 11 is the first message found in the file on the Integration CD at the following location:

..\Output_From_SPRINT\SON2_output.bin

The integration should now implement code that will correctly parse the multiplex data stream from the CP port. This can be verified by checking that the SON2 message that is decoded has similar data values to the messages that are being seen on the C1 debug port. It should be noted that as the SON2 message is ASCII the payload will not contain a DLE character, but other parts of the message (for example timestamp) might contain a DLE character that will be byte stuffed (see *Section 5.1*) so the parser should implement that logic.

7.4 Monitoring

This section introduces a number of methods of monitoring the status of a Lodestar/SPRINT instrument. This section is placed ahead of the remaining sections such that integrators are made aware of the features that are available to allow them to monitor for the correct effect of any changes that are being made.

7.4.1 BIST

The Built-In Self-Test (BIST) “Log” message is published at regular intervals of 2s. Other than the timestamp the rest of the message is a set of bit-fields with each bit representing whether a particular scenario/condition is present/active.

To assist with further integration the integrator should be able to decode the BIST message, indeed it should also be considered for use with any system that is utilising data from Lodestar/SPRINT as a health check of the instrument.

To configure the BIST message to be sent on the CP port, enter the following command:

```
LOG 0 MSG BIST
```

As the BIST is a binary message logging this message to the C1 debug port will not give a human readable output for comparison, however the following command can be entered that will print out the BIST to the terminal:

```
SYS BIST
```

It should be noted that the response to the command gives the current status of the BIST, whilst the bits in the message version are generally latched during the BIST update period.

To test that the BIST is being correctly decoded from the multiplex stream a number of the bits can be made to change state. If the integrator is following this guide as it is written the Lodestar/SPRINT is only running the AHRS, sending the following command will cause a reset to the AHRS algorithm:

```
GC RST
```

This should cause AHRS section bit 1 (“GCNotSettled”) to be set (to ‘1’) – see message specification for further details (*Section 8.3.6*). After a period of time the AHRS should settle and the same bit should be unset (‘0’) to indicate that the AHRS has settled. The number of seconds the AHRS will take to settle can found by interrogating the following configuration parameter:

```
GC SETTLE
```

7.4.2 Observation Status Messages

For every position, depth and velocity aiding input message a corresponding observation status message is generated. The observation status message details how that aiding message (observation) was used by the INS algorithm. Therefore an integrator should decode these messages to determine if the INS algorithm is using the observations being presented to it. If the INS isn't using the observations the observation status message will indicate why the observation has been rejected for use. It will not usually be necessary to decode all the observation status messages – only the ones that correspond to the aiding that will be used when integrated to the vehicle.

Whilst introduced in this section, the decoding of the observation status messages is best completed at the same time as implementing the input aiding.

7.4.3 Time System Status

Similar to the observation status messages, a Time System Status (TMS) message is available from a Lodestar/SPRINT that gives the current status of the instrument's time system. This should be decoded to determine if the time synchronisation being provided to the Lodestar/SPRINT is being accepted. It will also give values allowing an integrator the means to work out the current Lodestar/SPRINT relationship between Instrument Time and Common (UTC) Time.

If reading this guide in order the Lodestar/SPRINT is not being provided with any time synchronisation, however the TMS message will still be sent at regular intervals so message parsing can be checked at this point.

To enable the logging of the TMS message on the multiplex CP port enter the following command:

```
LOG 0 MSG + TMS
```

To aid in establishing if the decoding of the message is being completed correctly by the Lodestar/SPRINT the Instrument Time should be increasing in size when comparing one message to the next and the current value of UTC can be obtained by entering the following command:

```
TSYS DATETIME
```

7.4.4 General Logging

By (factory) default a Lodestar/SPRINT produces log files that contain a combination of “Log” and “Output” message types. The purpose of these log files is to allow offline processing to take place. This offline processing is typically completed using Sonardyne's Janus Application; this provides functionality to process DVL calibrations, fully reprocess navigation data and also allows sanity checking of a Lodestar/SPRINT instrument.

The Lodestar PC application gives a user an interface with which to upload the log files from the internal non-volatile memory, however an integrator may wish to consider creating their own log files. This is recommended if connecting the Lodestar PC application to the instrument is going to be problematic.

The log file format is simply a binary capture of all “Outputs” and “Logs” produced on a multiplex port – therefore very little processing is required other than reading the stream of data on a port from the instrument and saving to a file. The log files created internally are configured as follows:

```
OP SD MSG SON2 5.000

OP SD MSG + NAV 1.000 SRC 1

OP SD MSG + TEMP 0.500

LOG SD MSG CIMU NAVCAL NAVQUAL PMAT DXMAT TMS PRDSONDEPM PRDDPT PRDDIGIQM
PRDDIGIQPSI PRDDIGIQKPA PRDDIGIQO WINSON PRDSVX2DBAR PRDXDEPTH PRDKELLBIN
```

```
TRG CMD DVL LBL GPS PSONLBLBCN SVS PSONSS ALARM DBG TXT CPU UART SETTINGS
BIST SUSBL XPOS OBSTZMD OBSTGPSPOS OBSTSUSBL OBSTXPOS OBSTPDEPTH OBSTSVS
OBSTDVL OBSTLBL OBSTZUPT PWRSTAT
```

For an integrator to create similar log files the Lodestar/SPRINT should be configured for those “Outputs” and “Logs” above to be output on a port that is connected to the computer that is going to collate the data and create the log files. In this example this would be the vehicle control computer which would lead to the following configuration (in addition to any other “Outputs” or “Logs” that are required).

```
OP 0 MSG SON2 5.000

OP 0 MSG + NAV 1.000 SRC 1

OP 0 MSG + TEMP 0.500

LOG 0 MSG CIMU NAVCAL NAVQUAL PMAT DXMAT TMS PRDSONDEPM PRDDPT PRDDIGIQM
PRDDIGIQPSI PRDDIGIQKPA PRDDIGIQO WINSON PRDSVX2DBAR PRDXDEPTH PRDKELLBIN
TRG CMD DVL LBL GPS PSONLBLBCN SVS PSONSS ALARM DBG TXT CPU UART SETTINGS
BIST SUSBL XPOS OBSTZMD OBSTGPSPOS OBSTSUSBL OBSTXPOS OBSTPDEPTH OBSTSVS
OBSTDVL OBSTLBL OBSTZUPT PWRSTAT
```

Note

The port used should have a baudrate of at least 115200 to handle all of the outputs/logs listed above. Alternatively an Ethernet connection would provide ample bandwidth.

Whilst completing the rest of the steps in this guide it may not be worthwhile configuring all the messages above at this time; it is worth considering implementing all or a subset of the above towards the end of the integration effort.

7.5 Multiplex Input Message Aiding

This section will cover a number of different input messages that are used to aid the algorithms running on the Lodestar/SPRINT. The steps in this section should allow an integrator to configure the Lodestar/SPRINT and confirm that it is receiving correctly formed messages for the particular aiding source.

7.5.1 Time

The Lodestar/SPRINT can be time synchronised to UTC (or any time reference). This can be achieved by providing a 1PPS signal and ZDA NMEA-0183 message from a GPS receiver or similar clock source. If the serial communication link has known, stable latency then it is possible to use ZDA only, but 1PPS is recommended where possible. If external time synchronisation is lost the INS can continue to maintain an estimate of UTC time using its internal clock (~5ppm drift).

The ZDA message should conform to the NMEA 0183 standard. The message can be received on any channel or over Ethernet. The 1PPS signal should be a 5 V pulse with >1 µs duration. The signal can be fed to SPRINT via the CP+E1 or C1 port (trigger channels 1 or 2).

Note

***The Lodestar/SPRINT will accept a 3.3 V pulse but 5 V is recommended for signal integrity.**

Time synchronisation is not technically required for operation. However, it is still recommended to time synchronise the Lodestar/SPRINT when possible.

Time synchronisation with and without 1PPS is described below; if 1PPS is available it is suggested that time sync without 1PPS is completed first before integrating the 1PPS functionality.

7.5.1.1 Time Sync without 1PPS

The following assumes that the vehicle control computer connected to the CP port will be generating/or forwarding on the ZDA message to the Lodestar/SPRINT. The following commands will configure the Lodestar/SPRINT to receive and use a ZDA input on the CP port.

Note

Ensure that the vehicle computer is using the correct baud rate (115200 if using this manual) and is applying the multiplex protocol to the ZDA message.

```
IN 0 MSG + ZDA
TSYS SOURCE ZDA
TSYS ZDA 0
TSYS ZDALATENCY 0
```

The ZDA message should then be sent to the Lodestar/SPRINT wrapped in the multiplex protocol. To check that the ZDA message is being correctly received the following command can be entered:

```
LOG 1 MSG ZDA
```

The above command will result in ZDA messages being relayed on to the debug C1 port (together with the Log Wrapper; see *Section 5.2.3*) if the message format has been deemed correct. For the message to be deemed correct the multiplex wrapper is checked and decoded and then the format and contents of the ZDA message are checked. If the ZDA message is not being seen on the debug C1 port then one or more of the checks on the message are failing.

Further reading of this section is only required if it appears that the ZDA message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

If the ZDA messages are being rejected the message formation should be checked against the NMEA 0183 standard. To check that the ZDA message is correct without the multiplex format the following should be configured:

```
IN 0 MSG - ZDA
IN 1 MSG + ZDA
TSYS ZDA 1
```

Enter one or more ZDA messages on the debug C1 port (via the terminal), if a message is logged back then the ZDA format is correct, conversely if there is no logged message the format is incorrect and should be connected.

If the ZDA was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the ZDA to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the ZDA wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 0 MSG - ZDA
IN 1 MSG + ZDA
TSYS ZDA 0
OP 1 MSG 0
LOG 1 MSG 0
PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal. The same result could be achieved with a serial splitter.

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-2 ZDA Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 3D		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as an input to Lodestar/SPRINT Bit mask of 0x3000 to indicate the SID (0x3000 & 0x003D = 0x0000 = SID of 0) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x003D = 0x3D = 61)
4	24	$	Start character of ZDA message
5–39	47 50 5A 44 41 2C 31 31 30 36 34 38 2E 30 30 2C 32 33 2C 30 31 2C 32 30 31 37 2C 30 30 2C 30 30 2A 36 38	GPZDA,110648.00,23,01,2017,00,00*68	Rest of ZDA message
40–41	0D 0A	<cr><lf>	End of ZDA Message
42	52		Checksum
43	10		DLE (Data Link Escape)
44	03		ETX (End of Text)

Note

The example shown in *Table 7-2* is the first message found in the file on the Integration CD at the following location:

..\Input_to_SPRINT\ZDA.bin

7.5.1.2 Time Sync with 1PPS

This section is only applicable if a 1PPS is going to be provided as well as a ZDA message, it also assumes that the last section has been completed.

Assuming the 1PPS will be input on the input trigger of the CP/E1 connector (Trigger 2) the following commands should be entered:

```
TSYS SOURCE ZDA_1PPS
TSYS PPS 2
TSYS PPSMODE AFTER
```

The last command above is the configuration parameter that provides the relationship between the ZDA and 1PPS. The value of “AFTER” means that the ZDA is arriving after its associated 1PPS, this relationship is important for accurate time sync and therefore care should be taken to ensure that the correct setting is applied; for more detail on the options available; see *Section 7.6.13*.

To check that the 1PPS is being correctly received enter the following command:

```
LOG 1 MSG TRG
```

This will output a message to the debug C1 console each time a pulse is detected, if functioning correctly there should be one message generated per second.

```
$PSONTRG,0000919DB7EF,111705.000066,2,A,+,,*19
```

If there are too many or too few messages being generated the cabling of the 1PPS should be checked.

Note

If there are other triggers in the system that are active they will also be producing TRG messages at the same time.

In *Section 7.4.3* the TMS message was introduced and this should be used to determine whether the ZDA and 1PPS are being accepted by the Lodestar/SPRINT. Typically after 5 seconds with ZDA and 1PPS time sync the standard deviation of the UTC field should be less than $2x10^{-5}$ seconds. Also the Source field should indicate a ZDA and 1PPS update. Over time the other quality indicators (e.g. ZDA rejection count, PPS rejection count, etc.) can be checked to ensure the time sync is running well.

7.5.2 Position/Depth/Velocity

For a particular integration only a subset of the following subsections will be required to be implemented. Each section is independent so there is no need to read the subsections that are detailing aiding that will not be a part of the integration.

7.5.2.1 (S)USBL

There are two message types that are acceptable for (Subsea) USBL aiding, the first being a PSIMSSB message and the second being a modified GGA message.

PSIMSSB

To configure for an input of a PSIMSSB message the following command is required:

```
IN 0 MSG + PSIMSSB
```

To check the message and the multiplex wrapper are being accepted by the Lodestar/SPRINT the following command can be used to route a Log message to the debug C1 port.

```
LOG 1 MSG PSIMSSB
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Further reading of this section is only required if it appears that the PSIMSSB message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the PSIMSSB message is correct without the multiplex format the following should be configured:

```
IN 0 MSG - PSIMSSB
IN 1 MSG + PSIMSSB
```

Enter one or more PSIMSSB messages on the debug C1 port (via the terminal), if a message is logged back then the PSIMSSB format is correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the PSIMSSB was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the PSIMSSB to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the PSIMSSB wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 0 MSG + PSIMSSB
IN 1 MSG - PSIMSSB
OP 1 MSG 0
LOG 1 MSG 0
PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal. The same result could be achieved with a serial splitter.

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-3 PSIMSSB Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 98		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as an input to Lodestar/SPRINT Bit mask of 0x3000 to indicate the SID (0x3000 & 0x0098 = 0x0000 = SID of 0) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0098 = 0x98 = 152)
4	24	$	Start character of PSIMSSB message

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
5–73	50 53 49 4D 53 53 42 2C 31 33 31 39 32 33 2E 36 39 34 2C 42 30 31 2C 41 2C 2C 52 2C 4E 2C 4D 2C 30 2E 38 39 35 38 38 38 36 30 38 2C 2D 30 2E 30 31 34 35 38 35 33 33 33 2C 30 2E 30 2C 31 2C 4E 2C 2C 2A 37 38	PSIMSSB,131923.694, B01,A,,R,N,M,0.895888 608,- 0.014585333,0.0,1,N,,* 78	Rest of PSIMSSB message
74–75	0D 0A	<cr><lf>	End of PSIMSSB Message
76	E6		Checksum
77	10		DLE (Data Link Escape)
78	03		ETX (End of Text)

Note

The example shown in *Table 7-3* is the first message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Input_to_SPRINT\PSIMSSB.bin

GGA

To configure for an input of a SUSBL GGA message the following command is required:

```
IN 0 MSG + SUSBL
```

When providing a SUSBL GGA message it is important that the **SID** is correctly set (as the **MID** is the same as GPS GGA) to identify this as a message for SUSBL aiding. To check that the multiplexed wrapped SUSBL GGA message is being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG SUSBL
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Further reading of this section is only required if it appears that the SUSBL GGA message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the SUSBL GGA message is correct without the multiplex format the following should be configured

```
IN 0 MSG - SUSBL
IN 1 MSG + SUSBL
```

Enter one or more SUSBL GGA messages on the debug C1 port without the multiplex wrapper (via the terminal), if a message is logged back then the SUSBL GGA format is correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the SUSBL GGA was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the SUSBL GGA to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the SUSBL GGA wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 0 MSG + SUSBL
IN 1 MSG - SUSBL
OP 1 MSG 0
LOG 1 MSG 0
PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal. The same result could be achieved with a serial splitter.

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-4 SUSBL GGA Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–4	10 10 40		Byte 3 is an additional DLE byte due to the value of byte 2 (see *Section 5.1.1*) Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3000 to indicate the SID (0x3000 & 0x1040 = 0x1000 = SID of 1) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x1040 = 0x40 = 64)
5	24	$	Start character of SUSBL GGA message
6–81	47 50 47 47 41 2C 31 34 32 30 32 36 2E 37 36 31 2C 35 31 31 39 2E 38 33 38 37 32 2C 4E 2C 30 30 30 35 30 2E 31 34 30 36 34 2C 57 2C 35 2C 30 37 2C 31 2E 34 2C 30 2E 30 2C 4D 2C 30 2E 30 2C 4D 2C 32 2E 32 2C 30 33 36 32 2A 35 41	GPGGA,142026.761,5119.83872,N,00050.14064,W,5,07,1.4,0.0,M,0.0,M,2.2,0362*5A	Rest of SUSBL GGA message

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
82–83	0D 0A	<cr><lf>	End of SUSBL GGA Message
84	77		Checksum
85	10		DLE (Data Link Escape)
86	03		ETX (End of Text)

Note

The example shown in *Table 7-4* is the first message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Input_to_SPRINT\SUSBL_GGA.bin

7.5.2.2 GPS

Table 6-1 shows a number of NMEA messages are accepted by Lodestar/SPRINT. One of the messages listed under the GPS macro is the ZDA message and this has already been discussed in *Section 7.5.1*. In this section the GGA and VTG messages will be covered. The integration of any of the remaining messages shown in *Table 6-1* should be completed in consultation with Sonardyne. A GPS GGA message will be used to provide position information to the INS algorithm; in addition it will also be a source of input for the AHRS latitude compensation. The VTG message is only used by the AHRS algorithm as a source of velocity compensation. As the VTG is not used in INS processing there are no observation status messages produced for this aiding message. Velocity compensation of the AHRS is important for high speed vehicles or when operating at high latitude, but is not required if only INS outputs are used.

GGA

To configure for an input of a GPS GGA message the following command is required:

```
IN 0 MSG + GGA
```

When providing a GPS GGA message it is important that the SID is correctly set (as the MID is the same as SUSBL GGA) to identify this as a message for GPS aiding. To check that the multiplexed wrapped GPS GGA message is being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG GGA
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Note

Where `GGA` is used in the above command the macro name `GPS` could have been used, however by using the explicit message name the integrator will avoid confusion with any other messages that also fall under the macro.

Further reading of this section is only required if it appears that the GPS GGA message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the GPS GGA message is correct without the multiplex format the following should be configured:

```
IN 0 MSG - GGA
```

```
IN 1 MSG + GGA
```

Enter one or more GPS GGA messages on the debug C1 port without the multiplex wrapper (via the terminal), if a message is logged back then the GPS GGA format is correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the GPS GGA was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the GPS GGA to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the GPS GGA wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 0 MSG + GGA

IN 1 MSG - GGA

OP 1 MSG 0

LOG 1 MSG 0

PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal. The same result could be achieved with a serial splitter.

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-5 GPS GGA Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 40		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3000 to indicate the SID (0x3000 & 0x0040 = 0x0000 = SID of 0) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x1040 = 0x40 = 64)
4	24	$	Start character of GPS GGA message

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
5–80	47 50 47 47 41 2C 31 35 31 30 31 37 2E 30 39 37 2C 35 31 31 39 2e 38 33 38 39 35 2c 4e 2c 30 30 30 35 30 2e 31 34 31 37 36 2c 57 2c 35 2c 30 37 2c 31 2e 34 2c 30 2e 30 2c 4d 2c 30 2e 30 2c 4d 2c 32 2e 32 2c 30 33 36 32 2a 35 46	GPGGA,151017.097,5119.83895,N,00050.14176,W,5,07,1.4,0.0,M,0.0,M,2.2,0362*5F	Rest of GPS GGA message
81–82	0D 0A	<cr><lf>	End of GPS GGA Message
83	65		Checksum
84	10		DLE (Data Link Escape)
85	03		ETX (End of Text)

Note

The example shown in *Table 7-5* is the first message found in the file on the Integration CD at the following location:
..\Example_Communication_Files\Input_to_SPRINT\GPS_GGA.bin

VTG

To configure for an input of a VTG message the following command is required:

```
IN 0 MSG + VTG
```

To check that the multiplexed wrapped VTG message is being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG VTG
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Note

Where `VTG` is used in the above command the macro name `GPS` could have been used, by using the explicit message name the integrator will avoid confusion with any other messages that also fall under the macro

Further reading of this section is only required if it appears that the VTG message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the VTG message is correct without the multiplex format the following should be configured:

```
IN 0 MSG - VTG
IN 1 MSG + VTG
```

Enter one or more VTG messages on the debug C1 port without the multiplex wrapper (via the terminal), if a message is logged back then the VTG format is correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the VTG was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the VTG to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the VTG wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 0 MSG + VTG
IN 1 MSG - VTG
OP 1 MSG 0
LOG 1 MSG 0
PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal. The same result could be achieved with a serial splitter.

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-6 VTG Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–4	00 42		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0042 = 0x42 = 66)
5	24	$	Start character of GPS VTG message
6–46	47 50 56 54 47 2c 31 39 31 2e 33 2c 54 2c 31 39 31 2e 33 2c 4d 2c 30 30 30 2e 30 2c 4e 2c 30 30 30 2e 30 2c 4b 2c 44 2a 32	GPVTG,191.3,T,191.3,M,000.0,N,000.0,K,D*26	Rest of GPS VTG message
47–48	0D 0A	<cr><lf>	End of GPS VTG Message
49	69		Checksum
50	10		DLE (Data Link Escape)
51	03		ETX (End of Text)

Note

The example shown in *Table 7-6* is the first message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Input_to_SPRINT\VTG.bin

7.5.2.3 XPOS (External Position)

There are two possible means of entering a XPOS position, one via a message but it is also possible to enter a position update via a command. The generic multiplex command logic is covered in *Section 7.7* and should be read if the XPOS updates are going to be completed by command. It is recommended that the command method of XPOS update is only used for sporadic updates, if the updates will be regular in nature then it is recommended to use the XPOS message.

To configure for an input of a XPOS message the following command is required:

```
IN 0 MSG + XPOS
```

To check that the multiplexed wrapped XPOS message is being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG XPOS
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Further reading of this section is only required if it appears that the XPOS message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the XPOS message is correct without the multiplex format the following should be configured:

```
IN 0 MSG - XPOS

IN 1 MSG + XPOS
```

Enter one or more XPOS messages on the debug C1 port without the multiplex wrapper (via the terminal), if a message is logged back then the XPOS format is correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the XPOS was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the XPOS to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the XPOS wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 0 MSG + XPOS

IN 1 MSG - XPOS

OP 1 MSG 0

LOG 1 MSG 0

PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal. The same result could be achieved with a serial splitter.

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-7 XPOS Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 28		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0028 = 0x28 = 40)
4	24	$	Start character of XPOS message
5–60	58 50 4F 53 2C 31 32 30 39 30 39 2E 37 38 39 2C 35 31 31 39 2E 38 33 38 31 30 2C 4E 2C 30 30 30 35 30 2E 31 34 32 30 34 2C 57 2C 2C 2C 2C 30 2E 30 30 30 2C 2C 2A 32 43	XPOS,120909.789,5119.83810,N,00050.14204,W,,,,0.000,,*2C	Rest of XPOS message
61–62	0D 0A	<cr><lf>	End of XPOS Message
63	7C		Checksum
64	10		DLE (Data Link Escape)
65	03		ETX (End of Text)

Note

The example shown in *Table 7-7* is the first message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Input_to_SPRINT\XPOS.bin

7.5.2.4 LBL

For LBL aiding there are two message types that the SPRINT unit requires, these are PSONLOBS and PSONBCN (known as PSONLBLBCN for LBL). How these are generated from “raw” LBL communications is beyond the scope of this document.

LBL observations are recorded in a PSONLOBS message, however for that to be used a corresponding PSONBCN must have been recently received; this message contains details regarding the beacon. It is recommended that the two messages are sent as a pair, although it is possible to have the PSONBCN message being sent at a lower rate (compared to the PSONLOBS) and LBL aiding to still be possible.

Both messages are controlled with a message macro; therefore the following configuration should be applied to enable the input of LBL data:

```
IN 0 MSG + PSONLBLBCN
```

To check that both messages are being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG PSONLBLBCN
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Further reading of this section is only required if it appears that either the PSONLOBS or PSONLBLBCN messages are not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the messages are correct without the multiplex format the following should be configured

```
IN 0 MSG - PSONLBLBCN
IN 1 MSG + PSONLBLBCN
```

Enter one or more of each of the message types on the debug C1 port without the multiplex wrapper (via the terminal), if a message is logged back then the message formats are correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the messages were accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the LBL messages to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the LBL messages are wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 1 MSG - PSONLBLBCN
IN 0 MSG + PSONLBLBCN
OP 1 MSG 0
LOG 1 MSG 0
PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal. The same result could be achieved with a serial splitter.

The contents of this file should be inspected alongside the examples below to check that the multiplex wrapper is correct.

PSONLOBS

Table 7-8 PSONLOBS Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 A3		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00A3 = 0xA3 = 163)
4	24	$	Start character of PSONLOBS message
5–70	50 53 4F 4E 4C 4F 42 53 2C 32 32 32 2E 38 35 32 36 33 39 2C 32 34 30 32 2C 33 33 34 33 38 39 2E 30 30 30 2C 31 34 39 30 2E 30 30 30 2C 31 34 39 30 2E 30 30 30 2C 32 37 2C 30 2C 39 38 2C 41 2A 34 41	PSONLOBS,222.852639,2402,334389.000,1490.000,1490.000,27,0,98,A*4A	Rest of PSONLOBS message
71–72	0D 0A	<cr><lf>	End of PSONLOBS Message
73	95		Checksum
74	10		DLE (Data Link Escape)
75	03		ETX (End of Text)

PSONLBLBCN

Table 7-9 PSONLBLBCN Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 A0		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00A0 = 0xA0= 160)
4	24	$	Start character of PSONBCN message

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
5–71	50 53 4F 4E 42 43 4E 2C 32 32 32 2E 38 35 32 36 33 39 2C 32 34 30 34 2C 35 31 2E 33 32 39 38 37 33 38 38 2C 2D 30 2E 38 33 34 38 37 35 34 32 2C 31 30 36 2E 32 32 30 2C 38 30 2E 30 30 30 2C 30 2A 36 44	$PSONBCN,222.852639,2404,51.32987388,-0.83487542,106.220,80.000,0*6D	Rest of PSONBCN message
72–73	0D 0A	<cr><lf>	End of PSONBCN Message
74	B6		Checksum
75	10		DLE (Data Link Escape)
77	03		ETX (End of Text)

Note

The examples shown in *Table 7-8* and *Table 7-9* are the first messages found in the file on the Integration CD at the following location:
..\Example_Communication_Files\Input_to_SPRINT\LBL.bin

7.5.2.5 Pressure Depth (PRESS)

There are multiple message types that are accepted as a PRESS (Pressure Depth) input. These are listed in *Table 7-10* below.

Table 7-10 Supported PRESS sensor types

Message Name	MID	Description	Units
PRDDIGIQM	144	Paroscientific Digiquartz depth sensor	Metres of H_2O
PRDDIGIQPSI	158	Paroscientific Digiquartz depth sensor	PSI
PRDDIGIQKPA	159	Paroscientific Digiquartz depth sensor	kPa
PRDKELLBAR	142	Keller pressure sensor	Bar
PRDSONDEPM	145	Sonardyne depth message	Metres
PRDDPT	147	NMEA 0183 DPT message	Metres
WINSON	166	Tritech Winson Processed Data	PSI
PRDSVX2DBAR	168	Valeport Midas SVX2 Combined CTD/SVP	defined in message data (only DeciBar supported)
PRDXDEPTH	41	Sonardyne depth message (includes timestamp)	Metres

Note

The PRDKELLBAR support is currently specific to the integrated pressure sensor fitted to some Lodestar/SPRINT variants. An external Keller sensor could be used but would have to be configured to use the same device address and communications protocol as that used internally. Contact Sonardyne for details.

The following will detail two possible depth sensor inputs. The first will be PRDDIGIQPSI which will be configured such that it is connected directly to a Lodestar/SPRINT port, the second configuration will show how a PRDXDEPTH can be supplied via the vehicle control computer. These two variations should provide examples that can be applied to the other sensor types specified in *Table 7-10*.

Note

As most depth message formats are not timestamped, it is important to minimise the latency and jitter of the data. If the vehicle computer is likely to add significant latency or jitter it is recommended to connect the depth sensor directly to one of the Lodestar/SPRINT unit ports.

PRDDIGIQPSI

For the purposes of describing the configuration required for a direct connection to a pressure sensor it will be assumed that the pressure sensor is interfaced to the T1 port. The following will configure the T1 port to interface to a DigiQuartz pressure sensor, where the data is in PSI.

```
IN 3 MSG PRDDIGIQSPI
OP 3 BAUD 9600
OP 3 MULTIPLEX 0
OP 3 PROT 232
```

If the sensor is being independently powered then data should be provided to the Lodestar/SPRINT. If however the power pass through functionality of the Lodestar/SPRINT is being used the following command will turn the power on to the port (for more information on power pass through see *Sections 3.2.2* and *7.11.4*).

```
PORT 3 PWRPASS 1
```

To check that messages are being correctly sent, the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG PRDDIGIQSPI
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the configuration of the T1 port and the format of the message are correct. If this is not the case then the baud rate and other serial port configurations should be checked and the sensor itself can be checked by plugging the sensor directly into a PC serial port.

PRDXDEPTH

There are two possible means of entering a XDEPTH position, one via a message but it is also possible to enter a depth update via a command. The generic multiplex command logic is covered in *Section 7.7* and should be read if the XDEPTH updates are going to be completed by command. It is recommended that the command method of XDEPTH update is only used for sporadic updates, if the updates will be regular in nature then it is recommended to use the XDEPTH message.

To configure for an input of a XDEPTH message the following command is required:

```
IN 0 MSG + PRDXDEPTH
```

To check that the multiplexed wrapped XDEPTH message is being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG PRDXDEPTH
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Further reading of this section is only required if it appears that the XDEPTH message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the XDEPTH message is correct without the multiplex format the following should be configured:

```
IN 0 MSG - PRDXDEPTH
IN 1 MSG + PRDXDEPTH
```

Enter one or more XDEPTH messages on the debug C1 port without the multiplex wrapper (via the terminal), if a message is logged back then the XDEPTH format is correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the XDEPTH was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the XDEPTH to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the XDEPTH wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 1 MSG - PRDXDEPTH
IN 0 MSG + PRDXDEPTH
OP 1 MSG 0
LOG 1 MSG 0
PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal the same could be achieved with a serial splitter, which should achieve the same result (assuming there is no failure with the cable).

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-11 XDEPTH Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
2–3	00 29		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0029 = 0x29 = 41)
4	24	$	Start character of XDEPTH message
5–32	58 44 45 50 54 48 2C 31 31 31 39 33 34 2E 33 35 32 2C 30 2E 30 30 30 2C 2C 2A 32 45	XDEPTH,111934.352,0.000,,*2E	Rest of XDEPTH message
33–34	0D 0A	<cr><lf>	End of XDEPTH Message
35	79		Checksum
36	10		DLE (Data Link Escape)
37	03		ETX (End of Text)

Note

The example shown in *Table 7-7* is the first message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Input_to_SPRINT\XDEPTH.bin

7.5.2.6 DVL

Consistent latency and low jitter are critical for good DVL INS performance so DVL instruments are generally connected directly to the Lodestar/SPRINT and the scenario put forward in *Section 7.1* has the DVL connected to port T2.

Note

If the DVL is not connected directly to the Lodestar/SPRINT Sonardyne should be consulted for advice as to how to best integrate a DVL where DVL messages are routed via a vehicle control computer.

Lodestar/SPRINT support a number of different DVL message types as shown in *Table 7-12*.

Table 7-12 Supported DVL Messages

Message Type	LOG Message Name	MID	Description
PD0	DVLPD0	141	PD0, may be ASCII or Binary, provides BODY frame velocities
PD4	DVLPD4PD5	140	PD4, may be ASCII or Binary, provides BODY frame velocities
PD5	DVLPD4PD5	140	PD5, may be ASCII or Binary, provides BODY frame velocities (given the closeness to PD4 a PD5 message is handled using the same MID).

Message Type	LOG Message Name	MID	Description
ASONDV	DVLASONDV	227	Sonardyne Proprietary Message, Binary Only, provides BEAM velocities

Note

It is recommended that wherever possible Binary DVL messages are used with Lodestar/SPRINT. The use of binary messages over ASCII reduces the required bandwidth on the communications channels that are moving DVL data (the port used as an input as well as any that are logging the data).

When interfacing a DVL with Lodestar/SPRINT a decision needs to be made as to whether the DVL will be triggered by the Lodestar/SPRINT. The chosen configuration needs to be applied to both devices – if the Lodestar/SPRINT is configured for free-running (no trigger) but the DVL is setup for triggered operation the DVL will not produce any messages for the Lodestar/SPRINT. Conversely if the DVL is configured in a free running mode but the Lodestar/SPRINT is configured for triggered operation the integration could cause reduced performance as the Lodestar/SPRINT will be receiving DVL messages and it will not be able to corroborate those against the outgoing triggers.

The following configuration is assuming that the DVL being interfaced is a Sonardyne Syrinx instrument; advice can be sought from Sonardyne as to how this part of the configuration may change if another type of DVL is used. It is also assumed that the DVL is configured to expect a trigger. If the integration doesn't use a trigger *or* doesn't have the Lodestar/SPRINT trigger, the DVL trigger setup below can be skipped.

The following will configure the T2 port to expect Binary DVL messages:

```
IN 4 MSG DVL
DVL OPFORMAT BINARY
OP 4 MSG 0
OP 4 MULTIPLEX 0
OP 4 BAUD 115200
LOG 4 MSG 0
```

The following will configure the Trigger on the T2 port to trigger the DVL

```
DVL TRIG 4
TRIG 4 NRZ 0
TRIG 4 GO 1
```

If the sensor is being independently powered then data should be provided to the Lodestar/SPRINT. If however the power pass through functionality of the Lodestar/SPRINT is being used the following command will turn the power on to the port (for more information on power pass through; see *Sections 3.2.2* and *7.11.4*).

```
PORT 4 PWRPASS 1
```

To check that messages are being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG DVL
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the configuration of the T2 port and the format of the message are correct. If this is not the case then the baud rate and other serial port configurations should be checked. For the Sonardyne Syrinx this can be achieved via the web configuration (please refer to the Syrinx documentation for more details).

To check that the trigger is working correctly the following commands should be sent:

```
LOG 1 MSG + TRG
```

Assuming that there is only one message type defined for output on the DVL the output should alternate between logged DVL messages (which are usually binary) and a PSONTRG message.

Note

The assumption regarding one message type being sent by the DVL should be checked. Syrinx can be configured to output multiple types. As the Logging is specifying the macro name "DVL" all recognised messages will be logged which might make checking that the triggering is working correctly difficult. Where this is the case instead of using the message macro specific message name can be given, for example if it is known the DVL is producing PD4 messages the configuration should be changed to:

```
LOG 1 MSG TRG DVLPD4PD5
```

7.5.2.7 Sound Velocity/Speed (SVS)

There are a number of modes for configuring sound speed for the instrument. Two messages types are accepted and are detailed below, other possible command based options are covered in *Section 7.6.9*. The examples below gives details as to how a Valeport sensor would be configured if it was directly connected to a Lodestar/SPRINT communications port, the second gives an example of how a Sonardyne proprietary message can be configured via the vehicle command and control port.

VALEPORT

For the scenario it is assumed that the Valeport miniSVS sensor has been connected to the T1 port of the Lodestar/SPRINT unit and configured to "Factory" defaults, providing only SV data in Valeport Standard Format.

With this configuration on the sensor the following configuration would be required on the Lodestar/SPRINT:

```
OP 3 BAUD 19200
OP 3 MULTIPLEX 0
IN 3 MSG VALEPORT
SVS TYPE VALEPORT
SVS PORT 3
LOG 1 MSG VALEPORT
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then the sensor has been configured correctly, if no messages are seen on the C1 port the sensor configuration should be checked.

PSONSS

To configure for an input of a PSONSS message the following command is required:

```
IN 0 MSG + SVS
SVS TYPE PSONSS
```

```
SVS PORT 0
```

To check that the multiplexed wrapped PSONSS message is being correctly sent the following command will log correctly formatted messages to the debug C1 port:

```
LOG 1 MSG SVS
```

If all messages that are being sent to the Lodestar/SPRINT are being echoed on the debug C1 port then both the multiplex wrapper and the format of the message are correct.

Further reading of this section is only required if it appears that the PSONSS message is not being accepted – when any issues are resolved it is important to ensure that the configuration parameters are returned to the settings shown above.

To check that the PSONSS message is correct without the multiplex format the following should be configured:

```
IN 0 MSG - SVS

IN 1 MSG + SVS

SVS PORT 1
```

Enter one or more PSONSS messages on the debug C1 port without the multiplex wrapper (via the terminal), if a message is logged back then the PSONSS format is correct, conversely if there is no logged message the format is incorrect and should be investigated against the message specification.

If the PSONSS was accepted on the debug C1 port then the problem lies with the multiplex wrapper being used to send the PSONSS to the Lodestar/SPRINT on the CP port. Revert the configuration parameters by using the following commands and ensure that the PSONSS wrapped in the multiplex protocol is still being sent to the CP port:

```
IN 0 MSG + SVS

IN 1 MSG – SVS

SVS PORT 0

OP 1 MSG 0

LOG 1 MSG 0

PORT PASS 0 1
```

The last command routes all input bytes received on port “0” (CP) to the debug C1 port. The terminal will try and display the bytes in ASCII, but to investigate the multiplex header and footer the raw bytes being sent to the terminal should be logged to a binary file.

Note

In the steps above the PORT PASS functionality is used to route the raw communication being received to a terminal the same could be achieved with a serial splitter – which should achieve the same result (assuming there is no failure with the cable).

The contents of this file should be inspected alongside the example below to check that the multiplex wrapper is correct.

Table 7-13 PSONSS Multiplex Encoding

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 92		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0092= 0x92 = 146)
4	24	$	Start character of PSONSS message
5–28	50 53 4f 4E 53 53 2C 31 39 39 31 2E 30 30 2C 31 35 30 32 2C 4C 2A 34	PSONSS,1991.00,1502,M*4B	Rest of PSONSS message
29–30	0D 0A	<cr><lf>	End PSONSS Message
31	A6		Checksum
32	10		DLE (Data Link Escape)
33	03		ETX (End of Text)

Note

The example shown in

Table 7-13 is the first message found in the file on the Integration CD at the following location: ..Input_to_SPRINT\PSONSS.bin

7.6 Lodestar/SPRINT Configuration

7.6.1 IMU Configuration

7.6.1.1 IMU Lever Arms and Mounting Angles

For the purposes of configuring the Lodestar/SPRINT unit to correctly interpret aiding sensor data it is a requirement to know where the sensors that are providing that information are in relation to the Lodestar/SPRINT in 3-D space. This requires that a particular point on the vehicle is assigned to be the Common/Central Reference Point. Measurements then need to take place such that the X (Forward), Y (Starboard) and Z (Down) distances (Lever Arms) are known between the CRP and the Lodestar/SPRINT measurement point. Given that the Lodestar/SPRINT measurement point is in the middle of the unit it is best to use the drawings shown in *Section 3.6* to measure to a point on the external housing and then use the dimensions on the drawing to take into account where the Lodestar/SPRINT measurement point is in relation to the point used on the housing. When these distances are known they should be entered using the following command:

`IMU LA` *`X.X Y.Y Z.Z`*

e.g.

`IMU LA 1.5 0.5 -0.2`

The example shown above would mean the Lodestar/SPRINT measurement point is 1.5 m in front of the CRP, 0.5 m to starboard of the CRP and 0.2 above the CRP (note the Z axis here is defined as down so a negative number implies up).

Note

It is common to assume that the Lodestar/SPRINT measurement point is also the vehicle CRP – if this is the case no Lever Arms are require and can be left at zero, e.g.
`IMU LA 0.0 0.0 0.0`

In addition to the Lever Arms the other parameters that require configuration based on the mounting of the Lodestar/SPRINT on a vehicle are the mounting angles. *Section 4.3.2* details how the mounting angles should be calculated, this should give the following:

- Heading (Rotation by the gamma angle about Z axis)
- Resulting Pitch (Rotation by the beta angle about the resulting Y axis)
- Resulting Roll (Rotation by the alpha angle about the resulting x axis)

With the measurements and calculations completed the mounting angles can be entered using the following command:

`IMU MA` *`R.R P.P H.H`*

e.g.

`IMU MA 0.0 0.0 90.0`

The example above indicates that the Lodestar/SPRINT has been rotated by 90° around the Z axis, this type of mounting may be to allow easier cable routing. *Table 7-14* indicates the range of the angles that will be accepted for the command above (and all other commands that modify mounting angle parameters).

Table 7-14 Mounting Angle Input Limits

Angle	Minimum (degrees)	Maximum (degrees)
Alpha	-360	360
Beta	-90	90
Gamma	-360	360

Note

After modifying IMU Lever Arms or Mounting Angles, both the AHRS and INS algorithms should be reset.

7.6.2 GPS Aiding

GPS aiding is referring to GNSS position and GNSS velocity aiding. Explicitly excluded is time synchronisation using ZDA messages, described in *Section 7.5.1*.

7.6.2.1 GPS Lever Arms

Similar to the measurement of IMU Lever Arms described in *Section 7.6.1.1*, the distance from the vehicle CRP to the GPS antenna must be measured (in terms of forward, starboard and down). These distances should be entered into the system by using the following command:

```
GPS LA X.X Y.Y Z.Z
```

e.g.

```
GPS LA -0.5 0.03 -0.3
```

The example above indicates that the GPS antenna is 0.5 m aft of the CRP, 0.03 m to starboard of the CRP and 0.3 m above the CRP.

Note

After modifying GPS Lever Arms, the INS algorithm should be reset.

7.6.2.2 GPS Pre-filtering

For a GGA message to be used by the INS it is required that the Lodestar/SPRINT is time synchronised. When a GGA message is processed the internal time system is checked and based on the type of time sync being employed a decision is made as to whether the time system quality is sufficient to allow the GGA to be used. The quality parameter of the time system that is used is the time system standard deviation. When the Lodestar/SPRINT is configured to use ZDA and 1PPS time sync the time system standard deviation must be lower than 0.01s for the GGA to be accepted, when configured to just use ZDA as the time sync the standard deviation must be less than 0.6 s.

There is only one element of user configurable pre-filtering that takes place for GPS inputs and that is explicitly only on GGA messages. The GGA message reports a quality indicator after the latitude and longitude in the message. The following command allows the configuration of a minimum and maximum quality that will be accepted for use with in the INS:

```
GPS QUALITY min max
```

e.g.

```
GPS QUALITY 2 4
```

The minimum and maximum values can be the same value, for example:

```
GPS QUALITY 4 4
```

This has the effect of only accepting GGA messages with the quality “4”.

Note

The GPS QUALITY is only used for GGA rejection in the INS, for AHRS a GGA message will always be accepted for latitude aiding if quality field is be between 1 and 5 (inclusive), regardless of the setting above.

7.6.2.3 GPS INS Filtering/Configuration

The following command will instruct the INS on whether to use the incoming GGA data for a 2D or 3D position. When selected to “USEVERTICAL” the GGA data will be used to provide depth as well as position to the INS. This can be useful to correct for tidal changes during long periods at the surface. The following command would enable the GGA input for use as a 3-D position:

```
INS GPS USEVERTICAL 1
```

The following command allows the INS to filter based on the HDOP value contained within the GGA message (which is reported after the number of satellites). The value given to this parameter is the maximum allowed value of HDOP contained within the GGA message before the INS will reject the GGA observation. A value of 0 indicates that this filter/feature is disabled (as shown below):

```
INS GPS HDOPMAX 0.0
```

Note

There are further more advanced configuration parameters for tuning GGA aiding of the INS, where the default values are set such that in most cases modifications of the parameters will not be required. If problems are experience with GGA aiding or with modifying the above parameters please consult Sonardyne for further advice.

7.6.3 XPOS Aiding

7.6.3.1 XPOS Lever Arms

Similar to the measurement of IMU lever arms described in *Section 7.6.1.1,* the distance from the vehicle CRP to the XPOS measurement point must be measured (in terms of forward, starboard and down). These distances should be entered into the system by using the following command:

```
XPOS LA X.X Y.Y Z.Z
```

e.g.

```
XPOS LA -0.2 -0.3 0.3
```

The example above indicates that the XPOS transceiver is 0.2 m aft of the CRP, 0.3 m to port of the CRP and 0.3 m below the CRP.

Note

After modifying XPOS Lever Arms, the INS algorithm should be reset.

7.6.3.2 XPOS Pre-filtering

For XPOS messages to be used by the INS it is required that the Lodestar/SPRINT is time synchronised. When an XPOS message is processed, the internal time system is checked and based on the type of time sync being employed a decision is made as to whether the time system quality is sufficient to allow the XPOS message to be used. The quality parameter of the time system that is used is the time system standard deviation. When the Lodestar/SPRINT is configured to use ZDA and 1PPS time sync the time system standard deviation must be lower than 0.01 s for the XPOS message to be accepted. When configured to just use ZDA as the time sync the standard deviation must be less than 0.6 s.

XPOS can also be entered via a command. In this instance if a UTC time is provided the same check on the time system is made as above.

Other than checking of UTC time there are no other configurable parameters for pre-filtering XPOS observations.

7.6.3.3 XPOS INS Configuration

The following command allows an XPOS position to be entered via a command. This should be used only infrequently, if a more periodic XPOS position update is required a XPOS message should be configured for input; see *Section 7.6.3.3*.

```
INS XPOS 51.330900
```

The following command configures whether the INS will use the incoming XPOS for a 2D or 3D position. When selected to “USEVERTICAL” the XPOS message/command will be used to provide depth as well as position to the INS. The following command would enable the XPOS input for use as only a 2-D position.

```
INS XPOS USEVERTICAL 0
```

Note

There are further more advanced configuration parameters for tuning XPOS aiding of the INS, where the default values are set such that in most cases modifications of the parameters will not be required. If problems are experienced with XPOS aiding please consult Sonardyne for further advice.

7.6.4 SUSBL Aiding

7.6.4.1 SUSBL Lever Arms

Similar to the measurement of IMU Lever Arms described in *Section 7.6.1.1*, the distance from the vehicle CRP to the SUSBL beacon must be measured (in terms of forward, starboard and down). These distances should be entered into the system by using the following command:

```
SUSBL LA X.X Y.Y Z.Z
```

e.g.

```
SUSBL LA -0.2 -0.3 0.3
```

The example above indicates that the SUSBL beacon is 0.2 m aft of the CRP, 0.3 m to port of the CRP and 0.3 m below the CRP.

Note

After modifying SUSBL Lever Arms, the INS algorithm should be reset.

7.6.4.2 SUSBL Pre-filtering/Configuration

It is possible to filter which beacon number should be used for SUSBL aiding. The PSIMSSB should contain a beacon number, for a SUSBL GGA message the “Station ID” field can be utilised. The following command has two parameters, the first indicates whether filtering based on beacon ID should take place and the second is optional and gives the beacon ID that should be used if the filter is enabled.

```
SUSBL TPDR d [d]
```

The following example would disable the Beacon ID filter:

`SUSBL TPDR 0`

The following example would enable the filter and only observations from Beacon 2 will be used in processing:

`SUSBL TPDR 1 2`

The Time of Validity (TOV) that is attributed to the data in a SUSBL message (either GGA or PSIMSSB message) is determined by the time source mode. This is controlled by the following command:

`SUSBL TSOURCE` *`AUTO`* `|` *`TOA`* `|` *`TTAG`*

The `TOA` (Time of Arrival) option uses the time the message arrives at the Lodestar/SPRINT as the time of validity for aiding the INS. The `TTAG` (Time TAG) option utilises the UTC time field provided in the SUSBL message, for this mode the Lodestar/SPRINT must be time synchronised and the time system standard deviation must be below the value of the following parameter:

`SUSBL TSYNCSDLIM` *`d.d`*

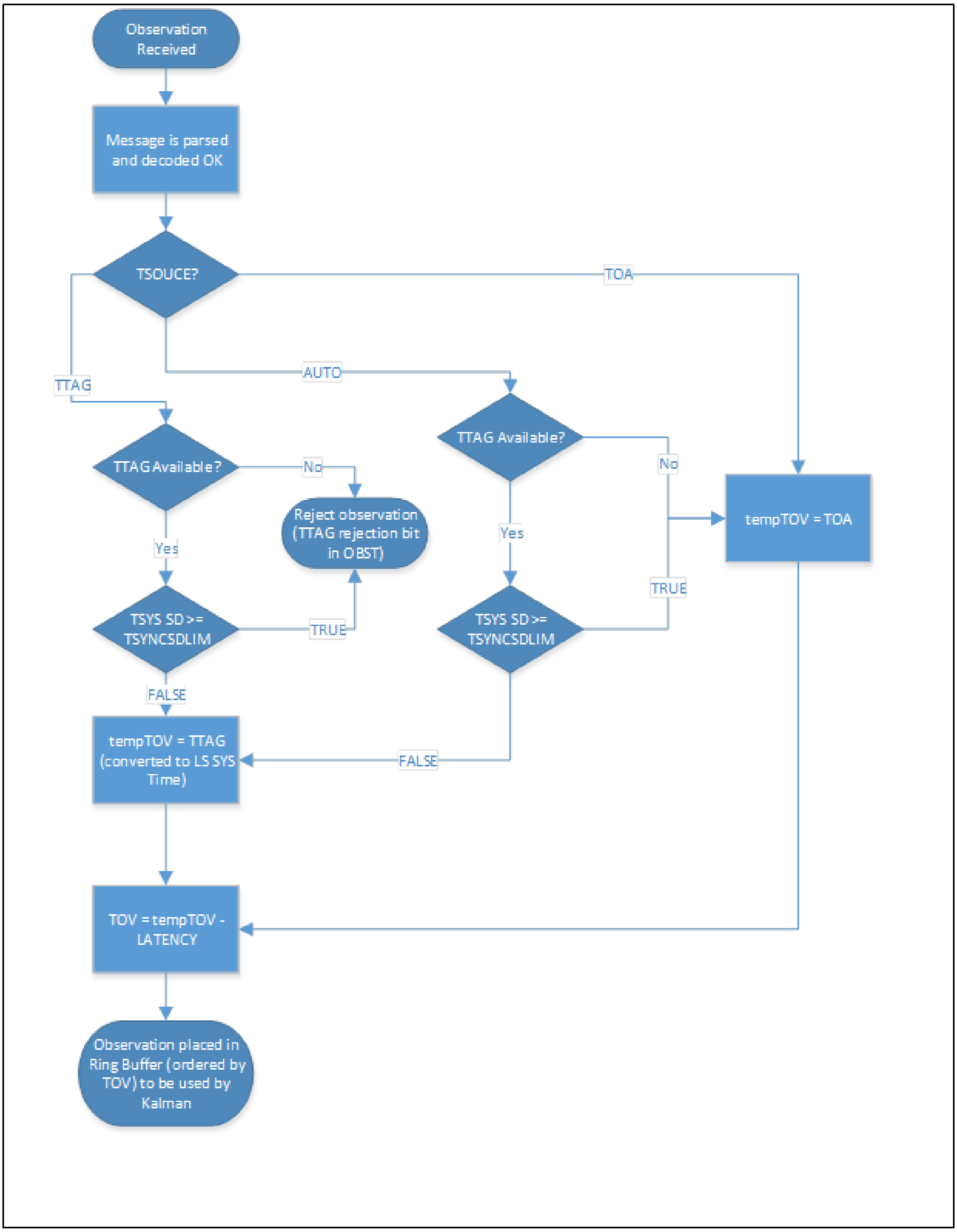
The `SUSBL TSOURCE AUTO` option will first try to use the time in the message (following the rules laid out for the `TTAG` option), if this check fails then the time of validity will default to that provided by the `TOA` option.

It is also possible to configure a fixed latency value for SUSBL observations; this can be controlled using the following command:

`SUSBL LATENCY` *`d.d`*

The relationship between the above three configuration parameters is illustrated in *Figure 7-1*.

Figure 7-1 SUSBL Time of Validity Source Selection



7.6.4.3 SUSBL INS Configuration

The following command configures whether the INS will use the incoming SUSBL data for a 2D or 3D position. When selected to “USEVERTICAL” the SUSBL data will be used to provide depth as well as position to the INS. The following command would enable the SUSBL input for use as only a 2D position:

```
INS SUSBL USEVERTICAL 0
```

Note

There are further more advanced configuration parameters for tuning SUSBL aiding of the INS, where the default values are set such that in most cases modifications of the parameters will not be required. If problems are experienced with SUSBL aiding please consult Sonardyne for further advice.

7.6.5 LBL Aiding

7.6.5.1 LBL Lever Arms

Similar to the measurement of IMU Lever Arms described in *Section 7.6.1.1*, the distance from the vehicle CRP and the LBL measurement point (TCVR fitted to the vehicle) must be measured (in terms of forward, starboard and down). These distances should be entered into the system by using the following command:

```
LBL LA X.X Y.Y Z.Z
```

e.g.

```
LBL LA -0.2 -0.15 -0.3
```

The example above indicates that the LBL measurement point is 0.2 m aft of the CRP, 0.15 m to port of the CRP and 0.3 m above the CRP.

Note

After modifying LBL Lever Arms, the INS algorithm should be reset.

7.6.5.2 LBL Pre-filter/Configuration

There are a number of different pre-filters that can be applied to LBL aiding; the following detail the parameters that are available for configuration.

The following command configures four parameters that are used to filter on the range provided by the PSONLOBS message:

```
LBL RANGE <d.d> <d.d> <d.d> <d.d>
```

e.g.

```
LBL RANGE 40.0 350.0 2.0 0.5
```

The first two parameter values are the minimum and maximum allowable ranges; if the range is outside of these limits the LBL observation is rejected. The third parameter value is the maximum range rate, with the fourth parameter value being the maximum value of the difference between the predicted and reported ranges. If either of the two final limits are exceeded by the range in the message the LBL observation is rejected.

A pre-filter is available that will only accept an LBL range observation for a beacon once a certain number of ranges have been collected for that beacon within a defined time period. This filter is configured using the following two commands:

```
LBL PASTOBSCNT <d>
```

```
LBL MAXTSINCEPASTTWT <d.d>
```

The first command configures the number of observations that must have been received from the beacon in the time configured with the second command:

e.g.

```
LBL PASTOBSCNT 2
```

```
LBL MAXTSINCEPASTTWT 21.0
```

The above configuration would mean the current LBL observation would only pass the filter if there had been two or more additional observations from the same beacon in the previous 21s.

The final LBL pre-filter command controls the pre-filtering based on the quality metrics found within the LBL observation message. The command syntax, together with an example are shown below:

```
LBL SIGNAL <d.d> <d.d> <d.d> <d.d> <d.d> [<d.d>]
```

e.g.

```
LBL SIGNAL -10.0 2.0 -12.0 3.0 30.0 50.0
```

The first parameter value (-10.0) controls the minimum allowable Signal to Noise Ratio (SNR) before the observation would be rejected. The third parameter (-12.0) sets the minimum limit on the reported Signal Level (SL) before the observation is rejected. The second parameter value (2.0) and fourth parameter value (3.0) controls the minimum filtered SNR and SL, with the fifth parameter value (30) providing the time constant to be used for the filtering (in seconds). The final (optional) parameter value is the minimum allowable cross correlation (XC) allowed before the observation is rejected.

7.6.5.3 LBL INS Configuration

Note

There are more advanced configuration parameters for tuning LBL aiding of the INS, where the default values are set such that in most cases modifications of the parameters will not be required. If problems are experienced with LBL aiding please consult Sonardyne for further advice

7.6.6 PRESS (Pressure Depth) Aiding

7.6.6.1 PRESS Lever Arms

Similar to the measurement of IMU Lever Arms described in *Section 7.6.1.1*, the distance from the vehicle CRP to the PRESS measurement point must be measured (in terms of forward, starboard and down). These distances should be entered into the system by using the following command:

```
PRESS LA X.X Y.Y Z.Z
```

e.g.

```
PRESS LA 0.2 0.3 -0.3
```

The example above indicates that the PRESS measurement point is 0.2 m forward of the CRP, 0.3 m to starboard of the CRP and 0.3 m above the CRP.

Note

After modifying PRESS Lever Arms, the INS algorithm should be reset.

7.6.6.2 PRESS Configuration

There are a couple of configuration parameters that are available for PRESS inputs. The first allows the user to enter a pressure offset (in metres) that is subtracted from the measured pressure before it is used in processing; this parameter is controlled using the following command:

```
PRESS OFFSET <d.d>
```

e.g.

```
PRESS OFFSET 1.0
```

The example above would result in 1 m being subtracted from the measured pressure.

The final parameter that can be configured is specific to when a Keller pressure sensor is being utilised. The parameter sets the measurement rate to be set and is controlled using the following command:

```
PRESS KELLRATE <d.d>
```

e.g.

```
PRESS KELLRATE 4.0
```

The example above would configure the Keller sensor to be interrogated for a measurement at a rate of 4 Hz.

7.6.6.3 PRESS (Pressure Depth) INS Filtering

Note

There are more advanced configuration parameters for tuning PRESS aiding of the INS, where the default values are set such that in most cases modifications of the parameters will not be required. If problems are experienced with PRESS aiding please consult Sonardyne for further advice

7.6.7 ZMD Aiding

7.6.7.1 ZMD Configuration

The following command allows the depth of the CRP used by the ZMD aiding to be configured:

```
ZMD CRPDETPH <d.d>
```

e.g.

```
ZMD CRPDEPTH 1.5
```

The example above indicates that at rest, the reported depth of the CRP will 1.5 m.

7.6.8 DVL Aiding

7.6.8.1 DVL Lever Arms and Mounting Angles

Similar to the measurement of IMU Lever Arms described in *Section 7.6.1.1* the distance from the vehicle CRP to the DVL measurement point must be measured (in terms of forward, starboard and down). These distances should be entered into the system by using the following command:

```
DVL LA <X.X> <Y.Y> <Z.Z>
```

e.g.

```
DVL LA 0.2 0.3 0.3
```

The example above indicates that the DVL measurement point is 0.2m forward of the CRP, 0.3m to starboard of the CRP and 0.3m below the CRP.

Similar to the measurement of IMU Mounting Angles described in *Section 7.6.1.1* the mounting angles of the DVL should be measured. These angles should be entered into the system by using the following command:

```
DVL MA <R.R> <P.P> <H.H>
```

e.g.

```
DVL MA 0.0 0.0 180.0
```

The example above indicates that the DVL forward mark has been rotated by 180 in heading when compared to vehicle forward.

Notes

DVL Mounting Angles are one of the parameters that are calculated as part of the DVL calibration. Where a DVL calibration has taken place the values from that calibration should be used (see *Section 7.11.8*). If a DVL calibration hasn't been carried out the mounting angles entered here only need to be coarse estimates before completing a DVL calibration.

After modifying DVL Lever Arms or Mounting Angles, the INS algorithm should be reset.

7.6.8.2 DVL Pre-filter/configuration

There are a number of different configuration and pre-filter parameters available for the DVL, some are only applicable when in certain modes of operation, where this is the case this is noted below.

The Lodestar/SPRINT can be set to use DVL observations in one of two modes: Beam or Body. Beam mode offers improved performance but is only available if the DVL is a Sonardyne Syrinx unit configured to output the Sonardyne proprietary ASONDV message. For all other message types and DVL manufacturers the Lodestar/SPRINT should be configured for Body mode using the following configuration parameter command:

```
DVL MODE BODY
```

Note

If a Syrinx unit is interfaced to the Lodestar/SPRINT and Beam mode of operation is required, Sonardyne should be consulted on the setting of configuration parameters.

Two further parameters that are calculated by a DVL calibration (as well as the DVL mounting angles) they are the DVL scale factor error and DVL latency. The scale factor parameter can be configured using the following command:

```
DVL SFERROR <d.d>
```

e.g.

```
DVL SFERROR 0.01
```

The example above would configure a scale factor error of 0.01 (1%) which will be used to scale the DVL data.

The DVL latency parameter can be configured using the following command:

```
DVL LATENCY <d.d>
```

e.g.

```
DVL LATENCY 0.1
```

The example above sets a latency of 0.1s to be applied when calculating the Time of Validity (TOV) for each DVL observation.

The Lodestar/SPRINT is able to receive either ASCII or binary versions of some DVL messages; see *Table 7-12*. The expected format of the incoming messages is controlled by the following command:

```
DVL OPFORMAT <ASCII | BINARY>
```

e.g.

```
DVL OPFORMAT BINARY
```

The example above would configure the Lodestar/SPRINT to expected only binary DVL messages.

Note

Lodestar/SPRINT do not support a mixture of Binary and ASCII DVL messages being sent on the same port.

For best performance it is recommended that the Lodestar/SPRINT is configured to trigger the DVL. This means that the DVL will acquire an observation whenever it receives a trigger from Lodestar/SPRINT. The time of validity of the observation will be accurately known by the Lodestar/SPRINT based on the trigger time. The following command enables/disables the trigger and configures which port to use:

```
DVL TRIG <T# | NONE>
```

e.g.

```
DVL TRIG NONE
```

The example above configures the Lodestar/SPRINT to not trigger the DVL and to use the Time of Arrival of messages as the basis for calculating the Time of Validity (TOV):

```
DVL TRIG 4
```

The example above configures the Lodestar/SPRINT to trigger a DVL on Trigger Port 4 (usually exposed on the T2 Connector). The trigger parameters can be further customised using the trigger commands documented in *Section 7.6.12*.

The following four configuration parameters control the pre-filtering of DVL observations:

```
DVL PREPMAXLAST <d.d>
```

This controls the allowable time period between the current and last DVL observations. If the time is greater than that configured, the current observation will not be used by the INS algorithm

e.g.

```
DVL PREPMAX LAST 1.2
```

The example above will reject the current DVL observation if the previous DVL observation is more than 1.2s older.

```
DVL PREPMAXACC <d.d>
```

This controls the allowable maximum change in velocity (acceleration) allowed as calculated between the current and previous DVL observations. If the velocity change is greater than that configured the current observation will not be used by the INS algorithm.

e.g.

```
DVL PREPMAXACC 0.25
```

The example above will reject the current DVL observation if the change in velocity exceeds 0.25 m/s^2.

```
DVL MINBEAMS <d> [<d.d>]
```

This controls the minimum number of DVL beams that must be valid in the message for the velocity to be used.

Note

The optional second value provided in the DVL MINBEAMS command is a “boost” factor which boosts the uncertainty of the observation before it is processed within the INS; this should be left at the default value unless otherwise advised by Sonardyne.

e.g.

```
DVL MINBEAMS 3
```

The example above would result in three beam solutions being used if four beam solutions are not available.

```
DVL KFEVEL <d.d>
```

This sets the maximum acceptable error velocity reported in the DVL observation. If the error velocity exceeds this value the observation will be rejected.

e.g.

```
DVL KFEVEL 0.01
```

The example above would result in a four beam solution being rejected if the error velocity is greater than 0.01 m/s.

Note

An error velocity will only be valid if the there are four valid beams, so the error velocity check will not be applied if there are only three, or fewer valid beams.

7.6.8.3 DVL INS Filtering

The following command controls filter constants used by the INS algorithm when using and estimating the DVL scale factor – the setting of this configuration parameter will be related to the robustness of the DVL calibration that has been undertaken; see *Section 7.11.8*.

```
INS DVL KFSF 0.002 1000
```

The example above shows that the filter's Bias/random noise, 1st order Markov RMS has been configured with a value of 0.002 and the filter's time constant has been set to 1000 s.

The following command controls filter constants used by the INS algorithm when using and estimating the DVL mounting angles – the setting of this configuration parameter will be related to the robustness of the DVL calibration that has been undertaken; see *Section 7.11.8*.

```
INS DVL KFMA 0.1 1000
```

The example above shows that the filter's Bias/random noise, 1st order Markov RMS has been configured with a value of 0.1 and the filter's time constant has been set to 1000 s.

Note

There are further more advanced configuration parameters for tuning DVL aiding of the INS, where the default values are set such that in most cases modifications of the parameters will not be required. If problems are experience with DVL aiding please consult Sonardyne for further advice.

7.6.9 SVS Aiding

7.6.9.1 SVS Configuration

The following command has already been introduced in *Section 7.5.2.7* where it was used to configure the two types of SVS input messages. It is also possible to configure SVS input to a further two modes that do not require SVS messages to be input to the Lodestar/SPRINT.

```
SVS TYPE AUTO
```

The above configuration utilises the Chen and Millero equation to calculate an estimate of sound velocity, this requires pressure (from pressure sensor), temperature (from DVL) and salinity (from configuration parameter); see *Section 7.6.9.2*.

```
SVS TYPE MANUAL
```

The above configuration uses a value configured by a parameter; see *Section 7.6.9.2*.

If there is no sound speed input the following should be used for the configuration parameter:

```
SVS TYPE NONE
```

7.6.9.2 SVS INS Configuration

The command below allows a manual sound speed to be entered into the system:

```
INS XSV 1500
```

The example above would set the sound speed to be used for DVL aiding to be 1500 m/s.

Note

The above command will return the latest value of sound speed available to the system, therefore if a value is not provided in the command it will return the current value being used by the Lodestar/SPRINT as has been previously entered (via Command or Message) or calculated in AUTO mode.

The following command sets the salinity of the sea water, as measured in parts per thousand and the value will be only be used (therefore required) if the SVS type is set to “AUTO”.

```
INS XSAL 32
```

7.6.10 ZUPT Aiding

7.6.10.1 ZUPT INS Configuration

The only configuration parameter for ZUPT aiding is to provide a limit on the expected velocities that would be encountered with ZUPT aiding enabled. This parameter is controlled by the following command:

```
ZUPT MAXVEL <d.d>
```

e.g.

```
ZUPT MAXVEL 0.001
```

The example above provides an expected maximum velocity of 0.001m/s for use by the INS algorithm when ZUPT aiding is enabled.

7.6.11 INS Configuration

7.6.11.1 INS Initialisation Prerequisites

For the INS to initialise it requires the following to be present:

- AHRS to have settled and be providing Attitude and Heading Data
- A timely position input which passes all of the filtering/rejection criteria and is configured for INS use
- A timely depth input which passes all of the filtering/rejection criteria and is configured for INS use

The BIST message contains a number of bits to assist in allowing a user to determine the status of the INS initialisation and what prerequisites may not be present. There is a top level bit to indicate if the INS is initialised together with bits to indicate if there is a suitable position and depth input available.

If aiding is believed to be being accepted by the SPRINT but the INS is not initialising the sections on INS Configuration (*7.6.11.2*) and Verifying Aiding Use (*7.6.11.4*) should be consulted.

7.6.11.2 INS Configuration

Aiding

The configuration parameter that is used to define what aiding is to be used by the INS is given below:

```
INS USE
```

To turn off all aiding to the INS algorithm the following command can be used:

```
INS USE 0
```

The following command will enable the INS algorithm to use SUSBL, DVL and PRESS as aiding sources:

```
INS USE DVL SUSBL PRESS
```

The example above assumes that SUSBL is not being used as a 3-D position. Assuming that at the time of entering the command the INS is not initialised the initialisation will take place when both position and depth observations that have passed their respective pre-filters and are provided to the INS algorithm (again assuming the AHRS has also settled).

Algorithm Configuration

Below is a non-exhaustive list of the configuration parameters and their corresponding commands that are used as configuration inputs to the INS algorithm. Commands not listed and discussed below should not be used unless advised by Sonardyne.

The following command will configure a value that will be used to boost the horizontal position covariance states when switching between two different position aiding sources. This tries to assist as in most cases there will be a difference between two position inputs due to their respective measurement errors, therefore the boost allows the INS to widen the position that it will accept when switching to a new source of position aiding. The command is as follows:

```
INS KFHPOSBOOST
```

To turn off the feature the configuration parameter can be set to 0 as follows:

```
INS KFHPOSBOOST 0
```

The following command configures a filter to be applied to the INS position which is used as the data for the output messages. This has the effect of “smoothing” the position produced by the INS algorithm over a certain period of time – this is to stop position “jumps” or “steps” being seen by a receiving system (these “jumps” or “steps” may have many causes but could be caused by a noisy SUBSL input). The command to configure these parameters is shown below:

```
INS RELNAVOUT 10 1 1
```

The first parameter is the time constant (in seconds) that will be used for the filter. The remaining two values give a maximum horizontal distance (second parameter) and maximum vertical distance (third parameter) in metres between the filtered position and the current INS position, if this limit is exceeded the filtered position is reset to the current INS position. To turn off this feature (as it is by default) the following configuration can be used:

```
INS RELNAVOUT 0 1 1
```

7.6.11.3 INS Reset (& AHRS Reset)

An INS reset can be triggered by entering the following command:

```
INS RST
```

This will cause the INS algorithm to restart and will therefore require the prerequisites set out in *Section 7.6.11.1* to be available for the INS algorithm to initialise again.

Should the Lever Arm or Mounting Angle configurations parameters be modified both an INS reset and AHRS reset should be undertaken by entering the following commands:

```
GC RST
INS RST
```

The INS is also reinitialised automatically if the INS predicted error grows above the value of the parameter that is configured via the following command:

```
INS KFHPOSRST
```

This is usually as the result of significant period of running on “free inertial”. This is where no aiding is provided/accepted for use by the INS algorithm and therefore the INS is running purely off the inertial sensors contained within the Lodestar/SPRINT. The default is set to 1km, but the time it will take for the error prediction to grow to that limit is difficult to predict as there are many factors that would influence the performance of the Lodestar/SPRINT during a period of free inertial. If the auto reset functionality is not required the following command will turn it off:

```
INS KFHPOSRST 0
```

7.6.11.4 Verifying Aiding Use by INS Algorithm

A simple way to keep track of whether the INS is initialised, which aiding sensors are being parsed and which sensors are being used by the INS can be seen by the PSONNAV message. The following command would setup the output on the C1 debug port (note for output messages to be seen command mode should be exited on the port):

```
OP 1 MSG PSONNAV 1
```

This configures a PSONNAV message to be output at 1 Hz, with the source set to AHRS. As some of the fields within the message can only be populated from the INS algorithm when the INS is initialised it will populate those fields, the sequence of PSONNAV messages below show this in practice:

```
<<$PSONNAV,104935.184,,,,,,,,V,,,0.196,-0.109,1.048,,A,Advs,,,,,*7D

<<$PSONNAV,104936.181,,,,,,,,V,,,0.196,-0.108,1.047,,A,Advs,,,,,*75

<<$PSONNAV,104937.180,,,,,,,,V,,,0.196,-0.109,1.047,,A,Advs,,,,,*74

<<$PSONNAV,104938.178,5119.835998,N,00050.141403,W,0.825,0.825,186.01,A,0
.002,0.668,0.196,-0.108,1.046,1.000,A,AIDvS,,,,,*32

<<$PSONNAV,104939.178,5119.835996,N,00050.141405,W,0.799,0.799,178.22,A,0
.004,0.609,0.197,-0.108,1.046,1.000,A,AIDvS,,,,,*3B

<<$PSONNAV,104940.176,5119.835994,N,00050.141406,W,0.711,0.711,143.69,A,0
.005,0.521,0.198,-0.107,1.046,1.000,A,AIDvS,,,,,*35
```

The first three messages show an output before the INS is initialised, with the final three showing that the INS has initialised as the position fields are now being populated.

The INS running status can also be determined from the “Sensor Status” field (for a full definition of the field contents the message definition should be consulted). For the first three messages this is “Advs”. A capital ‘A’ indicates that the AHRS algorithm is providing some data to the message (roll, pitch, heading), the remaining lower case letters indicate that aiding is being received but not used by the INS (‘d’ = depth, ‘v’ = DVL, ‘s’ = SUSBL). The final three messages show a change in the “Sensor Status” field with the addition of a capital ‘I’ indicating that the INS algorithm is providing data in the message output. Also two of the three sensor letters have changed from lowercase to upper case this indicates that some of the sensor data has been used by the INS in the previous INS cycle (‘D’ = depth, ‘S’ = SUSBL). However the DVL is still shown in lowercase (‘v’) – this could be because of two reasons:

1. The INS USE isn’t specifying the DVL for use, e.g.

```
INS USE SUSBL PRESS
```

2. The DVL is configured for use but is being rejected for use by the INS (either by the pre-filter or sigma rejection in the INS algorithm)

```
INS USE SUSBL PRESS DVL
```

Assuming that scenario 2 above is the configuration, the only way to determine why the DVL is not being used is to decode its corresponding Observation Status message (OBSTDVL). The tables below show how an OBSTDVL message should be decoded.

Note

Different DVL observations may be rejected for different reasons. Each OBSTDVL message only applies to one DVL observation.

Table 7-15 Multiplexed OBSTDVL Message

Byte Number	Byte Value (Hex)	Comment
0	10	DLE (Data Link Escape)
1	02	STX (Start of Text)
2–3	00 EB	Bit mask of 0x8000 to indicate if the time stamp is included – no timestamp (as this is included in the message payload for the OBSTDVL message) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00EB = 0xEB = 235)
4–47	16 5B A4 3B 02 00 70 00 00 00 00 00 00 50 A0 3B 02 00 00 00 C7 FB B7 44 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00	Payload
48	55	Checksum
50	10	DLE (Data Link Escape)
51	03	ETX (End of Text)

Note

The example shown in *Table 7-15* is the first message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Output_from_SPRINT\OBSTDVL.bin

Based on the message specification of the OBSTDVL message, *Table 7-16* decodes the message (after removing any byte stuffing) into its corresponding fields so that the reason for rejection can be found.

Table 7-16 OBSTDVL Message Field Decode

Field	Payload Byte Number	Byte Value (Hex)	Raw Value	Converted	Units
timeTag	0–5	16 5B A4 3B 02 00	9590561558	9590.561558	s
reject	6–7	70 00	0x0070	See *Table 7-17*	n/a
mahad	8–12	00 00 00 00	0	0	n/a
TOV	13–18	00 50 A0 3B 02 00	9590296576	9590.296576	s
GroupID	19	00	0	0	n/a
Obmsgtype	20	00	0	PD4	n/a
Sv	21–24	c7 fb b7 44	1471.8680419	1471.8680419	m/s
Residual1	25–28	00 00 00 00	0	0	m/s
Residual2	29–32	00 00 00 00	0	0	m/s
Residual3	33–36	00 00 00 00	0	0	m/s
Residual4	37–40	00 00 00 00	0	0	m/s
BeamReject	41–44	00 00 00 00	0	n/a	m/s

Table 7-17 provides a decoding of the “reject” field of the OBSTDVL message. This shows that there are three reasons this DVL observation has been rejected for use.

Table 7-17 OBSTDVL Rejection Bit Field Decode

Bit Number	Bit Name	Decoded Value
0	Not used	0
1	Not used	0
2	svs (SVS Bad)	0
3	cfg (DVL Configuration Bad)	0
4	evel (Reported Error Velocity above limit)	1
5	bsts (Bottom Status Bad)	1
6	zbr (Zero Beam Range detected)	1
7	zvel (Zero Velocity measured)	0
8	tout (Time between observations too large)	0
9	velcu (Velocity Change Unreasonable)	0
10	Not used	0
11	Not used	0
12	misc (Miscellaneous rejection)	0
13	ttag (Time Tag Issue)	0
14	sig (Kalman Sigma Rejected)	0
15	dis (Disabled from INS use)	0

7.6.12 Trigger Configuration

Some elements of Trigger functionality have already been covered (DVL trigger (*Section 7.5.2.6*) and 1PPS (*Section 7.5.1.2*). However the triggers can also be configured and used without having a specific function. A trigger can either be an input or an output, but when configured as an output the driven signal is also looped back into the system to allow the Lodestar/SPRINT to time stamp when the output occurred.

The following command configures a trigger to be either an input or an output:

```
TRIG X INPUT <1 | 0>
```

e.g.

```
TRIG 1 INPUT 1
```

The example above would configure trigger 1 as an input, the following example would configure trigger 4 as an output:

```
TRIG 4 INPUT 0
```

Note

The actual hardware of a Lodestar/SPRINT may only support either input or output functionality.

When configured as an input trigger the following command configures whether the active edge is the falling or rising edge.

```
TRIG X ACTIVE <1 | 0>
```

e.g.

```
TRIG 2 ACTIVE 1
```

The example above would configure trigger 2 as being rising edge triggered. To enable or disable an input trigger the following command should be used:

```
TRIG X GO <1 | 0>
```

e.g.

```
TRIG 2 GO 1
```

The example above would enable trigger 2.

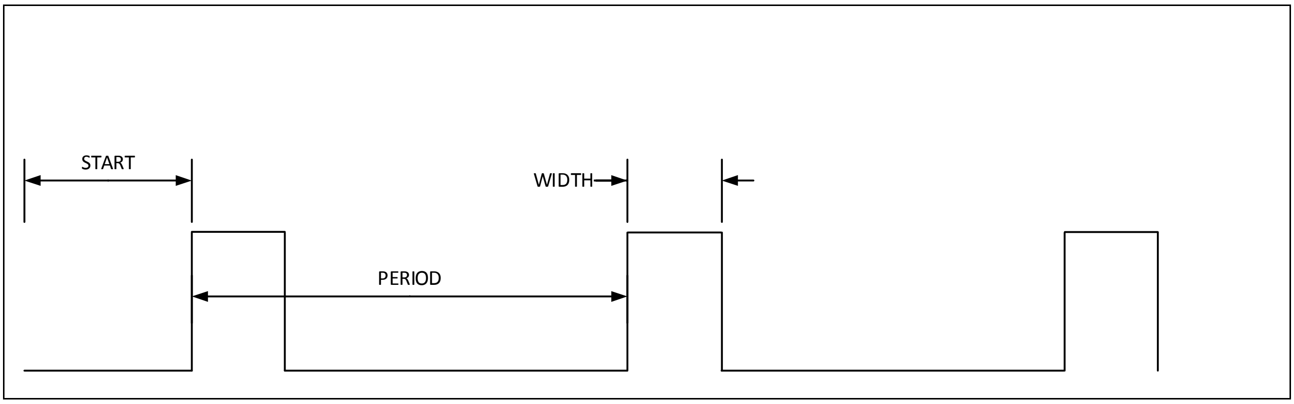
Note

Whenever a trigger configuration parameter is modified the trigger will require re-enabling using the 'GO' command.

When configured as an output trigger there are more extensive configuration options. The two configuration parameters that are described above for an input trigger are also valid for an output trigger. It should be noted that whenever a trigger configuration parameter is modified the trigger will require re-enabling (using the "GO" command). *Figure 7-2* provides the definition of the three additional configurable parameters available for an output trigger. The command syntax for the three commands is shown below:

```
TRIG X PERIOD <d.d>
TRIG X START <d.d>
TRIG X WIDTH <d.d>
```

Figure 7-2 Output Trigger Configuration Parameters




```
TRIG 3 ACTIVE 1
TRIG 3 START 50
TRIG 3 PERIOD 1000
TRIG 3 WIDTH 25
TRIG 3 GO 1
```

The above set of commands configures trigger 3 to have a period of 1000 ms where the width of the pulse to be 25 ms. The first rising edge is configured to be 50 ms after the trigger is enabled.

Note

The commands will ensure that erroneous configurations are not accepted, e.g. have a WIDTH that is larger than the configured PERIOD.

Depending on hardware support it may also be possible to configure an output trigger in “differential” mode; this is achieved by using the following command to modify the configuration:

```
TRIG X NRZ <0 | 1>
```

e.g.

```
TRIG 3 NRZ 0
```

The example above configures trigger 3 to not be a differential output.

Note

The above describes the most common configuration parameters; if more complex triggering architecture is required to be supported Sonardyne should be contacted for further support.

7.6.13 Time System Configuration

The majority of the time system configuration has already been discussed in *Section 7.5.1*. The following are additional configuration parameters and commands that control the time system.

The following command allows a ZDA latency parameter to be modified; this is only applied when only a ZDA message is being used to time synchronise:

```
TSYS ZDALATENCY <d.d>
```

e.g.

```
TSYS ZDALATENCY 0.1
```

The example above would apply a latency of 0.1s to the time within the ZDA message.

The following commands allow the Real Time Calendar (RTC) to be updated (or read). The RTC is used as a source of UTC when other time synchronisation is not available.

```
TSYS DATETIME <dd/mm/yyyy> <hh:mm:ss>
```

```
TSYS DATE <dd/mm/yyyy>
```

```
TSYS TIME <hh:mm:ss>
```

The above are three commands that allow the data and/or time to be updated on the RTC. The following command controls how often the RTC is updated if the Lodestar/SPRINT is being time synchronised by an external source (ZDA or ZDA and 1PPS):

```
TSYS UPDATE <d>
```

e.g.

```
TSYS UPDATE 150
```

The example above would result in the RTC being updated every 150 s with a UTC time based on the external time aiding being provided (assuming it has passed all the checks and is therefore classed as valid).

7.7 Multiplex Command and Control

Up until this point the C1 debug connection has been used to send commands to the Lodestar/SPRINT, however for some integrations there will be the need to control and modify configuration parameters via the vehicle control computer.

Commands on a multiplex port should be considered like any other input aiding that has been covered in the previous sections where the payload are the ASCII characters that up until this point in the document have been entered via a terminal program (on the debug/control port). The responses to commands are contained in one or more multiplex packets.

Contained on the Integration CD are two example files, one capturing the multiplex commands being sent to the Lodestar/SPRINT and one capturing the responses to those commands. The files can be found at the following locations:

..\Example_Communication_Files\Input_to_SPRINT\Commands.bin

..\Example_Communication_Files\Output_from_SPRINT\CMD_Responses.bin

The sequence of commands and responses found within these files is summarised below:

```
>>gc settle
<<GC SETTLE 120
<<ok
>>gc settle 360
<<GC SETTLE 360
<<ok
>>eng list
<<FIRMWARE 3.02.00.981 20170321102515
<<SYS LTL 325
<<SYS CMDS VERSION 116
…
<<PIC Application Status: SUBSEA_INS
```

```
<<PIC Application Running: ALL
<<PIC User Status: Factory
<<ok
>>enf list
<<not ok
```

The first command above provides an example of how a command can be used to interrogate the Lodestar/SPRINT to obtain the current value of a configuration parameter. *Table 7-18* and *Table 7-19* provide a walkthrough of the multiplex packets that combine for the first command sequence. It should be noted that the response is split across two multiplex packets, the first with the command response and the second with a report of command success.

Table 7-18 Multiplexed "GC SETTLE" Command

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 00		Bit mask of 0x8000 to indicate if the time stamp is included – should not be provided as this an input to Lodestar/SPRINT Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0000 = 0x00 = 0)
4	67	g	Start character of command
5–12	63 20 73 65 74 74 6C 65	c settle	Rest of command
13–14	0D 0A	<cr><lf>	End of command
15	3C		Checksum
16	10		DLE (Data Link Escape)
17	03		ETX (End of Text)

Table 7-19 Multiplexed Response to "GC SETTLE" Command

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	10 00		Bit mask of 0x8000 to indicate if the time stamp is included – for command responses this is true. Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0000 = 0x00 = 0)
4–9	16 D9 7F 2E 00 00		Timestamp in Lodestar/SPRINT Instrument Time
10	47	G	Start character of command response

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
11–22	43 20 53 45 54 54 4C 45 20 31 32 30	C SETTLE 120	Rest of command response
23–24	0D 0A	<cr><lf>	End of command response
25	31		Multiplex Checksum
26	10		DLE (Data Link Escape)
27	03		ETX (End of Text)
…With the command success packet following…			
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	10 00		Bit mask of 0x8000 to indicate if the time stamp is included – for command responses this is true. Bit mask of 0x3FF to indicate the MID (0x3FF & 0x0000 = 0x00 = 0)
4–9	3B D9 7F 2E 00 00		Timestamp in Lodestar/SPRINT Instrument Time
10–11	6F 6B	ok	Indication of command success
12–13	0D 0A	<cr><lf>	End of command response
14	30		Checksum
15	10		DLE (Data Link Escape)
16	03		ETX (End of Text)

The other commands that are included in the files provide further examples of commands and their respective responses. The next command modifies a configuration parameter successfully, this is followed by a command where its response is split over multiple lines, the final example shows an incorrect command where the response is only one multiplex packet indicating the command was “not ok”.

7.8 Multiplex Output and Log Messages

In *Section 7.3* an example AHRS message was generated and shown how it would be wrapped in the multiplex protocol. The following will provide examples for further output types including how to interpret binary output messages and those that are sourced from the INS algorithm and different remote points.

See *Section 8.3* for full details of the message contents.

7.8.1 INS Binary Message (LNAV)

The LNAV (Long NAVigation) message has been developed by Sonardyne to provide most (if not all) the data that is usually required for vehicle control and guidance, being binary it is also efficient with bandwidth (compared to ASCII messages conveying the same payload). An LNAV message was configured to be produced at 1 Hz with INS being the source of the information, as shown by the command below:

```
OP 0 MSG + LNAV 1 SRC 1
```

Table 7-20 shows how this message is wrapped in the multiplex format.

Table 7-20 Multiplexed INS LNAV message

Byte Number	Byte Value (Hex)	Comment
0	10	DLE (Data Link Escape)
1	02	STX (Start of Text)
2–3	00 E0	Bit mask of 0x8000 to indicate if the time stamp is included – no timestamp (as this is included in the message payload for the LNAV message) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00E0 = 0xE0 = 224)
4–94	8C 94 ED 10 10 00 00 BE E9 00 49 43 DE 67 FF 90 00 00 00 E7 03 22 00 EB FF 31 00 00 00 FF FF 04 00 00 00 00 00 00 00 32 00 7D 00 FC FF E4 F9 50 3E 7A 35 4F 3E 8A AF B3 43 72 2A 1A 3E 02 CD BA 3C 53 61 BB 3C B3 1C 7B 3F 17 13 59 3C 2E DB 4C 3C FE E2 B3 43 CE 51 18 3C 98 FF	Payload (note: Byte stuffing at Byte Number 8)
95	02	Checksum
96	10	DLE (Data Link Escape)
97	03	ETX (End of Text)

Note

The example shown in *Table 7-20* is the first message found in the file on the Integration CD at the following location:
..\Example_Communication_Files\Output_from_SPRINT\LNAV_INS.bin

The following table takes the byte stuffed payload from *Table 7-20* and removes the byte stuffing before indicating how the data should be decoded against the LNAV message specification.

Table 7-21 INS LNAV Message Decode

Field	Payload Byte Number	Byte Value (Hex)	Raw Value	Converted	Units
Time Tag	0–5	8C 94 ED 10 00 00	284005516	284.005516	s
Lat	6–9	BE E9 00 49	1224796606	51.3306328	deg
Lon	10–13	43 DE 67 FF	-9970109	-0.83568488	deg
Depth	14–17	90 00 00 00	144	0.144	m
Altitude	18–19	E7 03	999	9.99	m
Roll	20–21	22 00	34	0.186768	deg
Pitch	22–23	EB FF	-21	-0.115356	deg
Heading	24–25	31 00	49	0.269165	deg

Field	Payload Byte Number	Byte Value (Hex)	Raw Value	Converted	Units
vN	26–27	00 00	0	0	m/s
vE	28–29	FF FF	-1	-0.001	m/s
vD	30–31	04 00	4	0.004	m/s
wFwd	32–33	00 00	0	0	deg/s
wStbd	34–35	00 00	0	0	deg/s
wDwn	36–37	00 00	0	0	deg/s
aFwd	38–39	32 00	50	0.050	m/s^2
aStbd	40–41	7D 00	125	0.125	m/s^2
aDwn	42–43	FC FF	-4	-0.004	m/s^2
posMajor	44–47	E4 F9 50 3E	0.2040782570838928	0.2040782570838928	m
posMinor	48–51	7A 35 4F 3E	0.2023524343967437	0.2023524343967437	m
dirPMajor	52–55	8A AF B3 43	359.37140	359.37140	deg
stdDepth	56–59	72 2A 1A 3E	0.1505525410175323	0.1505525410175323	m
stdLevN	60–63	02 CD BA 3C	0.0228028334677219	0.0228028334677219	deg
stdLevE	64–67	53 61 BB 3C	0.0228735562413930	0.0228735562413930	deg
stdHeading	68–71	B3 1C 7B 3F	0.980906665325164	0.980906665325164	deg
velMajor	72–75	17 13 59 3C	0.01324918027967214	0.01324918027967214	m/s
velMinor	76–79	2E DB 4C 3C	0.01250342838466167	0.01250342838466167	m/s
dirVMajor	80–83	FE E2 B3 43	359.7733764648437	359.7733764648437	deg
velDown	84–87	CE 51 18 3C	0.00929684750735759	0.00929684750735759	m/s
status	88–89	98 FF	1111111110011000b	See *Table 7-22*	n/a

Table 7-22 INS LNAV Message Status Decode

Status Bit	Field Name	Value	Comment
0	bOrientationStatus	0b	OrientationStatus is valid
1	bPosStatus	0b	Position Status is valid
2	bAltitudeStatus	0b	Altitude field has new data
3	Not Used	1b	
4	bOrientationSource	1b	INS Supplying all data
5	bSubseaUSBLUsed	0b	SUSBL data has been used within last second
6	bDepthUsed	0b	PRESS data has been used within last second
7	bDVLUsed	1b	No DVL data has been used within last second

Status Bit	Field Name	Value	Comment
8	bLBLUsed	1b	No LBL data has been used within last second
9	bZUPTUsed	1b	No ZUPT data has been used within last second
10	bXPOSUsed	1b	No XPOS data has been used within last second
11	bGPSUsed	1b	No GPS data has been used within last second
12	bZMDUsed	1b	No ZMD data has been used within last second
13	bUSBLUsed	1b	No USBL data has been used within last second
14-15	Not Used	11b	

Note

The altitude is provided by the DVL and is provided even if the DVL observations are rejected for use by the INS algorithm, hence in the example above it is stated that the Altitude data is "new" but no DVL data has been used by the INS.

7.8.2 Remote Point $PSONNAV

The PSONNAV message is an ASCII message which contains many of the same fields as the LNAV; however for bandwidth considerations the precision on the data is constrained. The example given below is for the message to be output with respect to a remote point (see *Section 5.2.2.2*), the commands to configure the remote point and to configure the PSONNAV output are shown below:

```
SYS LA 3 3.0 4.0 -12.0
```

```
OP 0 MSG + PSONNAV 1 SRC 0 RP 3
```

The table below shows how this message is wrapped in the multiplex format.

Table 7-23 Multiplexed PSONNAV Message (based on Remote Point)

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	8C A9		Bit mask of 0x8000 to indicate if the time stamp is included – in this case it is Bit mask of 0x3C00 to filter the SID field (0x3C00 & 0x8CA9 = 0xC00, shifted gives RP = '3') Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00A9 = 0xA9 = 169)
4–9	60 81 C1 0C 00 00		Timestamp in Lodestar/SPRINT Instrument Time

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
10	24	$	Start character of PSONNAV message
11–74	50 53 4F 4E 4E 41 56 2C 31 30 32 38 31 30 2e 30 36 36 2C 2C 2C 2C 2C 2C 2C 2C 56 2C 2C 2C 30 2E 31 38 33 2C 2D 30 2E 31 31 37 2C 30 2E 39 31 31 2C 2C 56 2C 41 64 76 73 2C 2C 2C 2C 2C 2A 36 38	PSONNAV,102810.066,,,,,,,V,,,0.183,-0.117,0.911,,V,Advs,,,,,*68	Rest of PSONNAV message
75–76	0D 0A	<cr><lf>	End of PSONNAV message
77	66		Checksum
78	10		DLE (Data Link Escape)
79	03		ETX (End of Text)

Note

The example shown in *Table 7-23* is the first message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Output_from_SPRINT\PSONNAV_RP3.bin

7.8.3 BIST

The BIST message is a series of bit fields which provide a general overview of the health and status of the Lodestar/SPRINT.

The table below shows how this message is wrapped in the multiplex format. The payload is then used to demonstrate how individual bits of interest can be decoded.

Table 7-24 Multiplexed BIST Message

Byte Number	Byte Value (Hex)	Comment
0	10	DLE (Data Link Escape)
1	02	STX (Start of Text)
2–3	00 D9	Bit mask of 0x8000 to indicate if the time stamp is included – no timestamp (as this is included in the message payload for the BIST message) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00D9 = 0xD9 = 217)
4–52	CA 9E 39 1F 00 00 D5 03 00 00 02 00 03 00 00 00 00 00 00 00 00 00 F5 00 00 10 10 00 20 00 00 91 01 08 00 00 00 00 00 08 00 00 00 00 80 00 00 00 00	Payload
53	A9	Checksum

Byte Number	Byte Value (Hex)	Comment
54	10	DLE (Data Link Escape)
55	03	ETX (End of Text)

Note

The example shown in *Table 7-24* is the first message found in the file on the Integration CD at the following location:
..\Example_Communication_Files\Output_from_SPRINT\BIST.bin

The tables below give an indication how the BIST message should be decoded; first in to its composite fields and then down to bit level.

Table 7-25 BIST Field Decode

Payload Byte Number	Byte Value (Hex)	Field Name
0–5	CA 9E 39 1F 00 00	Time Tag
6–13	D5 03 00 00 02 00 03 00	Firmware Version
14–21	00 00 00 00 00 00 00 00	IMU
22–29	F5 00 00 10 00 20 00 00	Comms (see decode in *Table 7-26*)
30–37	91 01 08 00 00 00 00 00	CCA
38–39	08 00	AHRS
40–47	00 00 00 80 00 00 00 00	INS

Only when a bit field is equal to ‘1’ is the reason for the fields “active”. The table below decodes the “Comms” field in *Table 7-25* and presents the fields that are “active”.

Table 7-26 Active Bits in Comms Field

“Comms” Field Bit Number	Bit Field Name
0	**bUART0Rx**
2	**bUART2Rx**
4	**bUART4Rx**
5	**bUART0Tx**
6	**bUART1Tx**
7	**bUART2Tx**
28	**bSD**
45	**bPTP_Tx**

7.8.4 Settings

The Settings message is slightly different from the other messages already described in this section as, due to its size, it is transmitted over a number of multiplex messages. Where a message requires this behaviour it is stipulated in the message definition.

The Settings message is similar to the response to the `SYS LIST` command, it is output at a rate such that there is always one settings message logged in each SD card log file (the frequency of which is determined by the parameter controlled by the command `LOG ROTATE`). The first two bytes of the payload indicate how many multiplex packets containing fragments of the Settings message are required for a complete Settings message and an ID number to indicate which fragment of data is contained within the rest of the payload.

Table 7-27 Multiplexed Setting Message Fragment (First)

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	80 D8		Bit mask of 0x8000 to indicate if the time stamp is included – in this case it is Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00D8 = 0xD8 = 216)
4–9	7E DD E8 1C 00 00		Timestamp in Lodestar/SPRINT Instrument Time
10	56		Total number of fragments that make up the Settings message (total of 86)
11	01		Fragment ID (first fragment of Settings message)
12–107	53 59 53 20 47 59 52 4F 43 4F 4D 50 41 54 54 20 30 0D 0A 4F 50 20 53 44 20 4D 55 4C 54 49 50 4C 45 58 20 31 0D 0A 4F 50 20 53 44 20 4D 53 47 20 53 4F 4E 32 20 35 2E 30 30 30 0D 0A 4F 50 20 53 44 20 4D 53 47 20 2b 20 4E 41 56 20 31 2e 30 30 30 20 53 52 43 20 31 0D 0A 4F 50 20 53 44 20 4D	SYS GYROCOMPATT 0<cr><lf>OP SD MULTIPLEX 1<cr><lf>OP SD MSG SON2 5.000<cr><lf>OP SD MSG + NAV 1.000 SRC 1<cr><lf>OP SD	Rest of Settings message fragment
108	33		Checksum
109	10		DLE (Data Link Escape)
110	03		ETX (End of Text)

Note

The example shown in *Table 7-27* is the first message found in the file on the Integration CD at the following location:
..\Example_Communication_Files\Output_from_SPRINT\Settings.bin

Table 7-28 Multiplexed Setting Message Fragment (Last)

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	80 D8		Bit mask of 0x8000 to indicate if the time stamp is included–in this case it is Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00D8 = 0xD8 = 216)
4–9	7C FE AC 1D 00 00		Timestamp in Lodestar/SPRINT Instrument Time
10	56		Total number of fragments that make up the Settings message (total of 86)
11	56		Fragment ID (last fragment of Settings message)
12–96	32 35 0D 0A 2F 2F 20 53 59 53 20 43 4D 44 53 20 56 45 52 53 49 4F 4E 20 31 31 36 0D 0A 2F 2F 20 53 59 53 20 43 4D 44 53 20 4C 49 53 54 0D 0A 53 59 53 20 4E 45 54 20 31 39 32 2E 31 36 38 2E 31 37 39 2E 35 30 20 32 35 35 2E 32 35 35 2E 32 35 35 2E 30 0D 0A	25<cr><lf>// SYS CMDS VERSION 116<cr><lf>// SYS CMDS LIST<cr><lf>SYS NET 192.168.179.50 255.255.255.0<cr><lf>	Rest of Settings message fragment
97	39		Checksum
98	10		DLE (Data Link Escape)
99	03		ETX (End of Text)

Note

The example shown in *Table 7-28* is the last message found in the file on the Integration CD at the following location:
..\Example_Communication_Files\Output_from_SPRINT\Settings.bin

To recreate a completed Settings message all fragments are required with the payloads (minus the fragment count information) to be concatenated together.

7.9 Ethernet Connectivity

The previous sections have focussed on utilising serial communications as the primary means of passing data to and from the Lodestar/SPRINT. In most cases a TCP or UDP can be used to replace a serial port connection.

Note

Where an input message doesn't contain a timestamp indicating a time of validity it is advised that these are fed into the Lodestar/SPRINT via a serial connection to allow the Lodestar/SPRINT to internally tag the data with a time of arrival. Whilst this will also be completed for Ethernet based inputs, a direct serial connection by its nature will suffer less latency and jitter when compared to an Ethernet routed message.

7.9.1 Ethernet Input

In *Section 6.1,* one of the default TCP ports is configured for commands as follows:

```
IN 4001 NET TCP MSG COMMAND
```

Similar to how an input message would have been configured for a serial port the following would also add XPOS aiding to the port:

```
IN 4001 NET TCP MSG + XPOS
```

7.9.2 Ethernet Outputs and Logging

As with input message aiding, the command syntax and configurable items are similar to those used with the serial ports. The following shows an example configuration of an output message:

```
OP 4000 NET TCP MSG PSONNAV 1 SRC 1
```

7.10 Integrated Instrument Addendum

7.10.1 Combined Units Lodestar-Nav and SPRINT-Nav

Units may be interrogated to find out if they are combined units. Lodestar-Nav and SPRINT-Nav units will respond in the following way if they are a combined integrated instrument.

```
>>SYS SPRINTNAV
<<SYS SPRINTNAV 1
```

The above response indicates the unit is a Lodestar-Nav or SPRINT-Nav. This type of instrument brings together a Syrinx DVL with an integrated pressure (depth) sensor. The Syrinx DVL is connected to Port 4, for communications, power and trigger. The pressure sensor is connected to Port 2. The following configuration would configure the Lodestar/SPRINT to use the Syrinx DVL:

```
OP 4 BAUD 115200
OP 4 MULTIPLEX 0
OP 4 PROT 232
OP 4 MSG 0
IN 4 MSG DVL
LOG 4 MSG 0
DVL TRIG 4
PORT 4 PWRPASS 1
```

Notes

The above assumes the default configuration of the Syrinx DVL.

Even though the DVL is co-housed with the Lodestar/SPRINT the power to the DVL still needs to be commanded on (and off) via the Lodestar/SPRINT unless the DVL is being independently powered via the "DVL" connector.

The configuration for the internal pressure (depth) sensor is as follows:

```
OP 2 BAUD 9600
OP 2 MULTIPLEX 0
OP 2 PROT 485H
OP 2 MSG 0
LOG 2 MSG 0
IN 2 MSG PRDKELLBAR
```

Note

Power is provided to the pressure sensor as soon as power is supplied to the Lodestar/SPRINT.

7.11 Operational Considerations

7.11.1 Establishing a Connection to Lodestar/SPRINT

If it is possible that a Lodestar/SPRINT unit may be connected without being preconfigured correctly for the integration, the following steps should be followed by the Control Computer to establish a connection to the Lodestar/SPRINT before applying the configuration to the instrument.

The pseudocode below assumes a serial port connection as being the primary connection to the Lodestar/SPRINT. If the primary connection is actually a TCP connection, the loop that is trying different baud rates should be swapped for one that tries different IP addresses (if the Lodestar/SPRINT could have a non-default IP address) and TCP ports.

Figure 7-3 Steps for Establishing a Connection

Iterate over all supported baud rates

> *Send Multiplex Command to Lodestar/SPRINT (e.g. ENG LIST)*
>
> *If Multiplex reply received*
>
> > *Connection established – reconfigure Lodestar/SPRINT as required for integration – break from loop*
>
> *Else*
>
> > *Try to enter command mode on a non-multiplex port (see Section 6.1)*
> >
> > *If reply indicating entering command mode is received*
> >
> > > *Connection established – reconfigure Lodestar/SPRINT as required for integration – break from loop*
> >
> > *End If*
>
> *End If*

End Loop

7.11.2 Low Latency Outputs

The Lodestar/SPRINT instruments have been designed for the firmware to prioritise the integrity of the two algorithms (AHRS and INS) as such outputs do not take top priority in the firmware. Where this may become important is where there is a need for an output to have the lowest latency possible.

If this is the case, the recommendation is that a dedicated port be used for the message output. For best performance this should be a serial port. Within the firmware there is no message priority scheduling so for example an output will not jump ahead of a regular BIST message if the BIST message is already in an output queue.

Even when using a dedicated port it should not be assumed that on a message by message basis the period between messages is going to be consistent – for a dedicated output message @10 Hz over a period of time the average gap between messages will be close to 100 ms but gaps between any two messages can be smaller or larger than the 100 ms figure. Therefore if calculations are required on data that is included in the message and are time dependent it is advised to use the timestamp in the message to work out the delta time between the two messages for use in those calculations.

7.11.3 Aiding Switching

It is advised that the INS should be aided by only one position, one velocity and one depth input source at any one time. The following are examples where caution should be used in design of the logic that controls a SPRINT instrument:

- Velocity: Only one of the following should be enabled at a time: ZUPT and DVL
- Position: Only one of the following should be enabled at a time: GPS, SUSBL, XPOS, LBL and USBL
- Depth: It is possible for a number of the Position aiding sources to provide a 3D position which when configured for use would also provide a depth input to the INS. Therefore only one of the following should be enabled at a time: PRESS, ZMD, XPOS(3D), GPS(3D) and SUSBL(3D)

7.11.4 Power Pass Through

Section 3.2.2 shows a generic Lodestar/SPRINT is capable of controlling three different power outputs that are switched from the input supply. After each power on or reset the power pass through outputs are defaulted to being off. Therefore if they are required to be turned on to power external equipment, the integrator should ensure that they have the means to turn on the power pass through via a command, for example:

```
PORT 3 PWRPASS 1
```

When enabled the status of a power output can be obtained by inspecting the BIST message. For each power output there are two bits available for monitoring, the first indicates whether the power output is active, the second bit indicates if the output has tripped. In a tripped scenario a command needs to be sent to turn off the power output before a command to turn on the output will be actioned.

7.11.5 Communications Pass Through

The Lodestar/SPRINT can be configured such that communications can be routed from one port to another. This may be used to communicate to another instrument that is connected to the Lodestar/SPRINT.

It is down to the integrator to ensure that the bandwidth of both ports to be used is adequate for the amount of the data to be passed through between them. At the same time it is advisable to remove any IN, LOG or OP messages on the ports during the period of time where the communications pass through mode is going to be used.

To remove a port pass, the word “OFF” should be added to the end of the command, for example

```
PORT PASS 4 0
```

Would turn on passing through of communications between T2 and CP, the following command would turn off this communications path:

```
PORT PASS 4 0 OFF
```

7.11.5.1 Non-multiplex to Non-multiplex Pass Through

The configuration syntax is such that each setting sets up a unidirectional link, therefore two configuration items will need to be added for two way communications. For example:

```
PORT PASS 1 3
```

Will result in any data being received on Port 1 will be transmitted on Port 3.

```
PORT PASS 1 3
PORT PASS 3 1
```

Will result in any data being received on Port 1 will be transmitted on Port 3 as well as any data being received on Port 3 being transmitted on Port 1.

Note

If there are other “Outputs” or “Logs” configured on the output port configured for a Pass Through it should be expected that those “Output” or “Logs” will interrupt the data that is being passed through.

7.11.5.2 Multiplex to Non-multiplex Pass Through

In *Section 7.11.5.1* the two ports were unused and both configured to be non-multiplexed. In some cases it may necessary that data is passed between a non-multiplexed port and a multiplexed port.

An example might be the vehicle control computer (connected to CP – multiplexed) sending and receiving commands to a sensor directly connected to a port on the Lodestar/SPRINT, for example a DVL (connected to T2 non-multiplexed).

Table 7-29 and *Table 7-30* show an example of pass through communications, with a control computer requesting information from a Syrinx DVL connected on port T2. The following commands would enable the pass through for the following example:

```
PORT PASS 0 4
PORT PASS 4 0
```

Table 7-29 Multiplexed Pass Through Message (To Lodestar/SPRINT)

Byte Number	**Byte Value (Hex)**	**Equivalent ASCII Character(s)**	**Comment**
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)
2–3	00 DE		Bit mask of 0x8000 to indicate if the time stamp is included – input to so should not be present Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00DE = 0xDE = 222)
4–5	00 00		From (Source) Port = 0 = CP
6–7	04 00		To (Destination) Port = 4 = T2
8–15	0D 0A 50 4F 52 54 0D 0A	<cr><lf>PORT<cr><lf>	Pass through payload
16	C3		Checksum
17	10		DLE (Data Link Escape)
18	03		ETX (End of Text)

Note

The *example shown in Table 7-29 is the last message found in the file on the Integration CD at the following location:*

..*Example*_Communication_Files\Input_To_SPRINT\Multiplex_pass.bin

Table 7-30 Multiplexed Pass Through Message (From Lodestar/SPRINT)

Byte Number	**Byte Value (Hex)**	**Equivalent ASCII Character(s)**	**Comment**
0	10		DLE (Data Link Escape)
1	02		STX (Start of Text)

Byte Number	Byte Value (Hex)	Equivalent ASCII Character(s)	Comment
2–3	90 DE		Bit mask of 0x8000 to indicate if the time stamp is included – in this case it is Bit mask of 0x3C00 to indicate SID contents(0x3C00 & 0x90DE = 0x1000, shifting for SID = 4 which matches the “from” port) Bit mask of 0x3FF to indicate the MID (0x3FF & 0x00DE = 0xDE = 222)
4–9			
10–11	04 00		From Port = 4 = T2
12–13	00 00		To Port = 0 = CP
14–144	3E 3F 0D 0A 3E 50 4F 52 54 3A 50 49 44 30 2C 50 30 3B 42 52 31 31 35 32 30 30 2C 50 31 3B 42 52 39 36 30 30 2C 50 34 30 30 30 3B 54 43 50 2C 50 34 30 30 31 3B 54 43 50 2C 50 34 30 30 32 3B 54 43 50 2C 50 34 30 30 33 3B 54 43 50 2C 50 33 30 30 31 30 3B 55 44 50 2C 50 33 30 30 31 31 3B 55 44 50 2C 50 34 30 30 34 3B 54 43 50 3B 43 46 47 31 2C 50 33 30 30 31 32 3B 55 44 50 3B 43 46 47 31 0D 0A	>?<cr><lf>>PORT:PID0,P0;BR115200,P1;BR9600,P4000;TCP,P4001;TCP,P4002;TCP,P4003;TCP,P30010;UDP,P30011;UDP,P4004;TCP;CFG1,P30012;UDP;CFG1<cr><lf>	Pass through payload
145	B3		Checksum
146	10		DLE (Data Link Escape)
147	03		ETX (End of Text)

Note

The example shown in *Table 7-30* is the last message found in the file on the Integration CD at the following location:

..\Example_Communication_Files\Output_From_SPRINT\Multiplex_pass.bin

7.11.5.3 Multiplex to Multiplex Pass Through

This scenario is possible to configure, further advice should be sought from Sonardyne if the integrator determines that this may be a requirement of their system.

7.11.6 Providing Time Synchronisation from Lodestar/SPRINT

A Lodestar/SPRINT can be configured to output a 1PPS and corresponding $ZDA message. The 1PPS is generated on one of the output trigger channels whilst the corresponding $ZDA message can be output on one or more ports. Even if only a $ZDA message is required the use of an output trigger channel is still required.

The $ZDA message generated is valid for the corresponding previous 1PPS. In this example the trigger on port T1 (trigger 3) is not used and therefore could be adopted for the 1PPS generation, the following gives an example configuration where a $ZDA message would also be sent to C1.

```
LOG 1 MSG LSZDA
TSYS LSZDA 1 3 0
```

The 1PPS and $ZDA generation are based on the firmware's idea of Common Time (UTC). Where there is no external time synchronisation provided to the Lodestar/SPRINT, the firmware uses its Instrument Time and its relationship to the Common Time (UTC) reference from the RTC. Where the Lodestar /SPRINT is being externally time synchronised the firmware will utilise its Instrument Time as well as its continually updating relationship with Common Time (UTC) for generation.

7.11.7 Communication and CPU Loading Checks

The Lodestar/SPRINT actively checks when changing the configuration that there is sufficient communications bandwidth on ports for the configured or to be configured "Outputs" and "Logs". This check is also completed when a serial port baud rate is requested to be reduced. Therefore the integrator should ensure that such commands are accepted, if they are not then attention should be paid to whether all "Outputs" and/or "Logs" are required. In addition for "Outputs" a reduction in output rate could also be considered.

In addition to the communications bandwidth described above which is completed on a port by port basis, the Lodestar/SPRINT will also check the CPU loading of the configured or to be configured "Output" and "Logs". Similar to the communication bandwidth checks if the Lodestar/SPRINT thinks that the CPU will be overloaded the configuration will not be allowed.

A snapshot of the current UART and CPU load can be achieved by the following commands:

```
SYS UART
SYS CPU
```

7.11.8 DVL Calibration

It is recommended that to calibrate a DVL if one is fitted to the vehicle and is being used to aid the INS. For a calibration the vehicle will need to undertake a number of manoeuvres and the corresponding logged data from the Lodestar/SPRINT; see *Section 7.4.4*. This data then is required to be post-processed (offline) in the Sonardyne Janus utility, an output of which will be a DVL calibration report, as shown in *Figure 7-4*.

Figure 7-4 DVL Calibration Report

Results:

DVL params	a (roll) [º]	b (pitch) [º]	g (heading) [º]	Scale factor error [%]	Latency [s]
Before	-0.050	-0.006	0.160	-0.035	-0.001
Calculated	**0.461**	**-0.033**	**0.327**	**-0.113**	**0**
Calculated Accuracy	0.067	0.014	0.066	0.010	

The results shown in *Figure 7-4* would then need configuring on the Lodestar/SPRINT using the following commands:

```
DVL MA 0.461 -0.033 0.327
DVL LATENCY 0
DVL SFERROR -0.00113
```

Depending on the calibration that has taken place a further two configuration parameters should be configured for the INS.

For an uncalibrated DVL (heading good to 3°) the following configuration should be used:

```
INS DVL KFSF 0.025 300
INS DVL KFMA 2.5 300
```

For a calibrated DVL with position being derived from USBL the following configuration should be used:

```
INS DVL KFSF 0.005 300
INS DVL KFMA 0.5 300
```

For a calibrated DVL with position being derived from RTK GPS the following configuration should be used:

```
INS DVL KFSF 0.002 1000
INS DVL KFMA 0.1 1000
```

Note

The SPRINT User Manual and Janus User Manual should be consulted for further information with regards to performing the DVL Calibration.

7.11.9 IMU Output

A CIMU message is provided that allows delayed, compressed IMU data to be logged for offline processing to take place. Depending on the license that has been granted it may be possible to also configure the 'IMU' message for output. This message gives the angular rates and accelerations together with their Time of Validity (TOV); however the output of this message will still be delayed by 60s. A further extension to the license can allow "Real-Time" output, which will allow the IMU message to be output without the imposed 60s delay (the output will still be affected by small amounts of latency and jitter depending on port loading and port type).

Note

Some output messages also contain angular rate and acceleration information, these are provided for vehicle control but due to filtering and interpolation/extrapolation they should not be considered to be the equivalent of IMU data.

Section 8 – Messages

This section contains message specifications for the following message types:

8.1 Message IDentifiers (MIDs)

Table 8-1 Message Identifiers

Message Name	MID	Definition	Notes
NMEA GGA	64	*8.2.11*	
NMEA INGGA	105	*8.2.11*	GGA produced from SPRINT INS algorithm
NMEA VTG	66	*8.2.12*	
NMEA ZDA	61	*8.2.10*	Also includes “LSZDA”
NMEA DPT	149	*8.2.3*	
PRDXDEPTH	41	*8.2.6*	$XDEPTH
PRDSONDEPM	145	*8.2.2*	$SONDEP
PRDDIGIQM	144	*8.2.1*	Digiquartz Pressure Sensor Report (in metres)
PRDDIGIQPSI	158	*8.2.1*	Digiquartz Pressure Sensor Report (in PSI)
PRDDIGIQKPA	159	*8.2.1*	Digiquartz Pressure Sensor Report (in kPa)
PRDSVX2DBAR	168	*8.2.5*	Midas SVX Depth (in Bar)
PRDKELLBIN	233	*8.3.4*	Processed Pressure and Temperature data from a Keller Pressure Gauge
WINSON	166	*8.2.4*	Tritech Winson data (for depth aiding)
PD4/PD5	140	*8.2.7*	DVL
PD0	141	*8.2.8*	DVL
ASONDV	227	For internal Sonardyne use only	For logging only
CIMU	218	For internal Sonardyne use only	For logging only
IMU	203	Definition available depending on licensing and instrument type	Inertial Measurement Unit output
LNAVUTC	232	*8.3.25*	Long Navigation message (UTC timestamped)
SETTINGS	216	n/a	ASCII output similar to a “SYS LIST”
TMS	208	*8.3.5*	Time System data
LNAV	224	*8.3.25*	Long Navigation message
BIST	217	*8.3.6*	Built In Self Test
NAVCAL	204	*8.3.8*	INS Nav Data

Message Name	MID	Definition	Notes
NAVQUAL	214	*8.3.6*	INS Navigation Quality
PMAT	205	For internal Sonardyne use only	For logging only
DXMAT	207	For internal Sonardyne use only	For logging only
PHMAT	229	For internal Sonardyne use only	For logging only
PSIMSSB	152	*8.2.15*	Subsea USBL aiding
SVS	143	*8.2.17*	Valeport Sound Speed
PSONSS	146	*8.2.16*	Other Sound Speed
CMD	512	n/a	Logging of all command across all ports (ASCII)
TRG	110	*8.3.7*	Trigger Event Log
DBG	92	n/a	Debug Text (ASCII)
TXT	92	n/a	Info Text (ASCII)
CPU	248	n/a	CPU Loading (ASCII)
UART	248	n/a	UART Loading (ASCII)
PWRSTAT	248	n/a	Power Status (ASCII)
PSONLBLBCN	160	*8.2.14*	LBL aiding
PSONLOBS	163	*8.2.13*	LBL aiding
OBSTZMD	170	*8.3.2.2*	ZMD Observation Status
OBSTGPSPOS	172	*8.3.2.3*	GPS (GGA) Observation Status
OBSTSUSBL	174	*8.3.2.4*	SUSBL Observation Status
OBSTXPOS	175	*8.3.2.5*	XPOS Observation Status
OBSTPDEPTH	176	*8.3.2.6*	PRESS Observation Status
OBSTSVS	177	*8.3.2.7*	SVS Observation Status
OBSTDVL	235	*8.3.2.8*	DVL Observation Status
OBSTLBL	179	*8.3.2.9*	LBL Observation Status
OBSTZUPT	180	*8.3.2.10*	ZUPT Observation Status
NAV	213	*8.3.10*	Navigation message
PRDID	24	*8.3.11*	Orientation message
TSS1	4	*8.3.12*	Orientation message
TSS2	5	*8.3.13*	Orientation message
PSONNAV	165	*8.3.24*	ASCII Navigation message
XPOS	40	*8.2.18*	External Position aiding
TSS3	6	*8.3.14*	Orientation message
EM1000	8	*8.3.15*	Orientation message
EM3000	10	*8.3.16*	Orientation message
PHTRO	20	*8.3.17*	Orientation message

Message Name	MID	Definition	Notes
HDT	2	*8.3.18*	Orientation message
THS	3	*8.3.19*	Orientation message
TEMP	212	*8.3.20*	Temperature
SON2	120	*8.3.21*	Orientation message
POSMV111	31	*8.3.22*	Orientation message
POSMV113	32	*8.3.23*	Orientation message

8.2 IN Messages

8.2.1 Digiquartz Pressure Sensor Report

Description

Pressure depth output from Paroscientific Digiquartz intelligent pressure depth sensor.

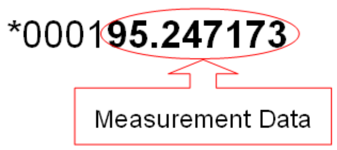
Format

*0001nnn__<cr><lf>

Digiquartz Formatting

Field	Description
*	Start character
00	Destination ID (00 is ID of serial host)
01	Source ID (01 is ID of sender device)
nnn__	Measurement Data **(units=Metres H2O/KPA/PSI)**
<cr><lf>	return plus linefeed

Supported Input Format



8.2.2 $PSONDEP Report

Description

Proprietary depth input string.

Format

$PSONDEP,x.xx,y.y,c*hh<cr><lf>

PSONDEP Formatting

Field	Description
$	Start_character
PSONDEP	Header
x.xx	Depth
y.y	Observation Error
c	Units (M=metres)
*hh	Terminator and Checksum
<cr><lf>	Termination (0x0D 0x0A)

Supported Input Format

8.2.3 $__DPT REPORT

Description

This is NMEA string outputs depth below water surface.

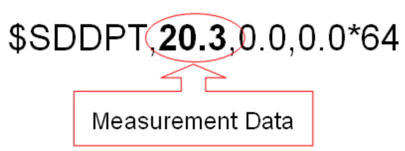
Format

$__DPT,x.x,y.y,z.z*hh<cr><lf>

DPT Formatting

Field	Description
$	Start_character
__DPT	Header
x.x	Water depth in metres
y.y	Offset in metres. (NOT USED)
z.z	Maximum range scale in use
*hh	Terminator and Checksum
<cr><lf>	Termination (0x0D 0x0A)

Supported Input Format



8.2.4 Tritech Winson Processed Data

Description

This message is generated by some Tritech Bathymetric systems. The message can be used to provide depth aiding to Lodestar AAINS. The message contains a number of measurements but only the depth data is used.

Format

%DBBBBSSRRAPmTTTTTPPPPPPPPPPmTTTTTPPPPPPPPPPRRRRRRRRRRmOOOOOCCCCCmTTTTTSSSSSVVVVVmAAAAAAAAAADDDmDDDDDDDDDDHHMMSSCC<cr><lf>

Winson Data

Field	Description
%D	Start characters
BBBB	Number of bytes in message, displayed as hex-ascii
SS	Slot number, displayed as hex-ascii ('01' to '0C')
RR	Generic device type, displayed as hex-ascii (00 to 63)
A	Data reply mode (0=ASCII, 1=Hex, 2=Binary, 3=CSV)
P	Send type (0=Processed, 1=Raw, 2=SeaKing Short, 3=SeaKing Long)
m	'+' if positive, '-' if negative
TTTTT	Internal temperature
PPPPPPPPPP	Digiquartz pressure
m	'+' if positive, '-' if negative
TTTTT	Digiquartz temperature
PPPPPPPPPP	Raw digiquartz pressure reading is the number of 8MHz counts for 10,000 digiquartz pulses
RRRRRRRRRR	Raw digiquartz temperature reading is the number of 8MHz counts for 40,000 digiquartz pulses
m	'+' if positive, '-' if negative
OOOOO	Local oscillator calibration coefficient
CCCCC	Conductivity
m	'+' if positive, '-' if negative
TTTTT	Conductivity probe temperature
SSSSS	Salinity (calculated from conductivity readings)
VVVVV	Velocity of sound (calculated from conductivity readings)
m	'+' if positive, '-' if negative
AAAAAAAAAA	Altimeter reading (This value DOES include 'Altimeter Position offset'. V.O.S. figure in preceding field is applied)

Field	Description
DDD	Bathymetric system devices, byte converted to Uint Bit 0 = 1 = Digiquartz valid Bit 1 = 1 = Conductivity valid Bit 2 = 1 = Altimeter valid Bit 3 = 1 = Internal temperature valid (Only installed in SK701 Bathy) Bit 4 = 1 = Velocity of sound calculation valid Bit 5 = 1 = Salinity calculation valid E.G. Digiquartz valid = “001” Digiquartz & Conductivity valid = “003” Digiquartz & Altimeter valid = ”005” Digiquartz, Conductivity & Altimeter valid = “007”
m	‘+’ if positive, ‘-‘ if negative
DDDDDDDDDD	Depth (This value DOES include ‘Bathy Position offset’ and ‘Bathy Zero offset’)
HHMMSSCC	Time at start of scan
<cr><lf>	Terminator, return plus linefeed

Supported Input Format

```
%D0074042700-
1023000933736 79+015490002885627000233 2115+0000045425+013903835215176+0000
001639055+000062725201580003<cr><lf>
```

8.2.5 Midas SVX2 Depth

Description

This message is tab delimited and provides Sound Velocity, Depth, Temperature and conductivity, however only depth information is used.

Format

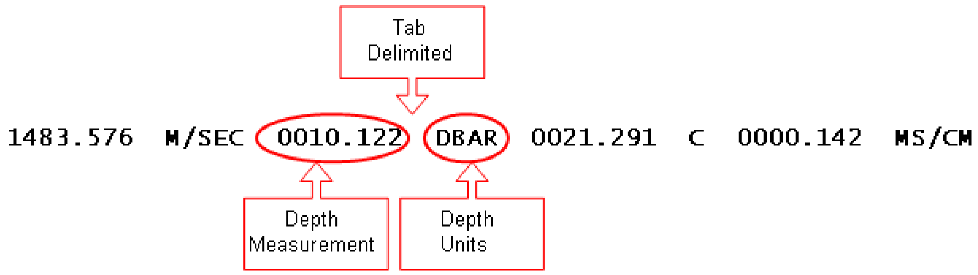
ssss.sss<tab>uuuu<tab>dddd.ddd<tab>xxxx<tab>tttt.ttt<tab>xxxx<tab>cccc.ccc<tab>zzzz<cr><lf>

SVX2 Data

Field	Description
ssss.sss	Sound Velocity in metres per second
uuuu	Sound Velocity Units (M/SEC)
dddd.ddd	Depth
xxxx	Depth Units (DBAR)
tttt.ttt	Temperature
xxxx	Temperature Units (C)
cccc.ccc	Conductivity
zzzz	Conductivity Units (MS/CM)

Field	Description
<cr><lf>	return plus linefeed

Supported Input Format



8.2.6 XDEPTH

Description

External depth input from a non-specific source.

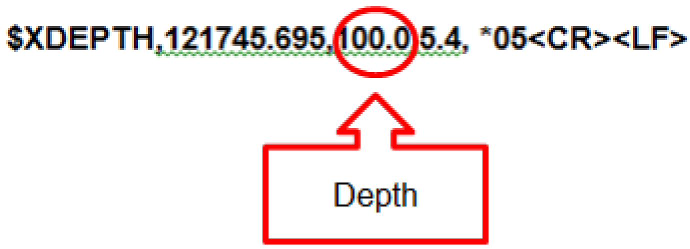
Format

$XDEPTH,hhmmss.sss,d.ddd,x.xxx,aa*hh<cr><lf>

XDepth Formatting

Field	Description
$XDEPTH	Start Character
hhmmss.sss	UTC Time
d.ddd	Depth in metres
x.xxx	Depth Standard Deviation in metres
aa	Spare
<cr><lf>	return plus linefeed

Supported Input Format



8.2.7 PD4

Table 8-2 PD4

Hex Digit	Binary Byte	Field	Description
1,2	1	DVL Data ID	Stores the DVL (speed log) identification word (7Dh).
3,4	2	Data Structure	Identifies which data pattern will follow based on the Pdcommand. 0 = PD4 = Bytes 1 through 47. 1 = PD5 = Bytes 1 through 45 and bytes 46 through 88 from PD5 table
5–8	3,4	No. of Bytes	Contains the number of bytes sent in this data structure, not including the final checksum.
9,10	5	System Config	Defines the DVL hardware/firmware configuration. Convert to binary and interpret as follows. BIT 76543210 00xxxxxx BEAM-COORDINATE VELOCITIES 01xxxxxx INSTRUMENT-COORDINATE VELOCITIES 10xxxxxx SHIP-COORDINATE VELOCITIES 11xxxxxx EARTH-COORDINATE VELOCITIES xx0xxxxx TILT INFORMATION NOT USED IN CALCULATIONS xx1xxxxx TILT INFORMATION USED IN CALCULATIONS xxx0xxxx 3-BEAM SOLUTIONS NOT COMPUTED xxx1xxxx 3-BEAM SOLUTIONS COMPUTED xxxxx010 300-kHz DVL xxxxx011 600-kHz DVL xxxxx100 1200-kHz DVL **Note: Field used for Syrinx identifier:** **xxxxx111 Syrinx DVL**
11–14	6,7	X-Vel Btm	These fields contain the velocity of the vessel in relation to the bottom in mm/s. Positive values indicate vessel motion to east (X), north (Y), and up (Z). LSD = 1 mm/s
15–18	8,9	Y-Vel Btm	
19–22	10,11	Z-Vel Btm	
23–26	12,13	E-Vel Btm	
27–30	14,15	Bm1	These fields contain the vertical range from the ADCP to the bottom as determined by each beam. This vertical range does not compensate for the effects of pitch and roll. When a bottom detection is bad, the field is set to zero. Scaling: LSD = 1 centimetre; Range = 0 to 65535 cm
31–34	16,17	Bm2 Rng to	

35–38	18,19	Bm3 Bottom	
39–42	20,21	Bm4	
43,44	22	Bottom Status	This field shows the status of bottom-referenced correlation and echo amplitude data. Convert to binary and interpret as follows. A zero code indicates status is OK. BIT 76543210 1xxxxxxx BEAM 4 LOW ECHO AMPLITUDE x1xxxxxx BEAM 4 LOW CORRELATION xx1xxxxx BEAM 3 LOW ECHO AMPLITUDE xxx1xxxx BEAM 3 LOW CORRELATION xxxx1xxx BEAM 2 LOW ECHO AMPLITUDE xxxxx1xx BEAM 2 LOW CORRELATION xxxxxx1x BEAM 1 LOW ECHO AMPLITUDE xxxxxxx1 BEAM 1 LOW CORRELATION
45–48	23,24	X-Vel Ref Layer	These fields contain the velocity of the vessel in relation to the water-mass reference layer in mm/s. Positive values indicate vessel motion to east (X), north (Y), and up (Z). LSD = 1 mm/s
49–52	25,26	Y-Vel Ref Layer	
53–56	27,28	Z-Vel Ref Layer	
57–60	29,30	E-Vel Ref Layer	
61–64	31,32	Ref Layer Start	These fields contain the starting boundary (near surface) and the ending boundary (near bottom) of the water-mass reference layer (BL-command). If the minimum size field is zero, the ADCP does not calculate reference-layer data. Scaling: LSD = 1 dm; Range = 0-9999 dm
65-68	33,34	Ref Layer End	
69,70	35	Ref Layer Status	This field shows the status of reference layer depth and correlation data. Convert to binary and interpret as follows. A zero code indicates status is OK. BIT 76543210 xxx1xxxx ALTITUDE IS TOO SHALLOW xxxx1xxx BEAM 4 LOW CORRELATION xxxxx1xx BEAM 3 LOW CORRELATION xxxxxx1x BEAM 2 LOW CORRELATION xxxxxxx1 BEAM 1 LOW CORRELATION
71,72	36	TOFP Hour	These fields contain the time of the first ping of the current ensemble.
73,74	37	TOFP Minute	
75,76	38	TOFP Second	
77,78	39	TOFP Hundredth	
79-82	40,41		BIT Results These fields contain the results of the ADCP’s Built-in Test function. A zero code indicates a successful BIT result. BYTE 40 BYTE 41 (BYTE 41 RESERVED FOR FUTURE USE)

			1xxxxxxx xxxxxxxx = RESERVED x1xxxxxx xxxxxxxx = RESERVED xx1xxxxx xxxxxxxx = RESERVED xxx1xxxx xxxxxxxx = DEMOD 1 ERROR xxxx1xxx xxxxxxxx = DEMOD 0 ERROR xxxxx1xx xxxxxxxx = RESERVED xxxxxx1x xxxxxxxx = DSP ERROR xxxxxxx1 xxxxxxxx = RESERVED
83-86	42,43	Speed of Sound	Contains either manual or calculated speed of sound information (EC-command). Scaling: LSD = 1 metres per second; Range = 1400 to 1600 m/s
87-90	44,45		Temperature Contains the temperature of the water at the transducer head. Scaling: LSD = 0.01 C; Range = -5.00 to +40.00 C
91-94	46,47	Checksum	This field contains a modulo 65536 checksum. The ADCP computes the checksum by summing all the bytes in the output buffer excluding the checksum. Note: This field contains the checksum only when the PD4-command is used.

8.2.8 PD0

General Format

Bytes	Field name	Notes
6 + [2 × No. of data types]	HEADER	Always output
59	FIXED LEADER DATA	
65	VARIABLE LEADER DATA	
2 + 8 per depth cell	VELOCITY	WD-command WP-command
2 + 4 per depth cell	CORRELATION MAGNITUDE	
2 + 4 per depth cell	ECHO INTENSITY	
2 + 4 per depth cell	PERCENT GOOD	
85	BOTTOM TRACK DATA	BP-command
2	RESERVED	Always output
2	CHECKSUM	

Header Format

Byte#	Field name	Notes
1	Header ID	Stores the header identification byte (7Fh)
2	Data source ID	Stores the data source identification byte (7Fh for the Work-Horse ADCP)
3-4	Number of bytes in ensemble	This field contains the number of bytes from the start of the current ensemble up to, but not including, the 2-byte checksum
5	Spare	Undefined
6	Number of data types	This field contains the number of data types selected for collection. By default, fixed/variable leader, velocity, correlation magnitude, echo intensity, and percent good are selected for collection. This field will therefore have a value of six (4 data types + 2 for the Fixed/Variable Leader data).
7–8	Offset for data type #1	These field contains the internal memory address offset where the Navigator will store information for data type #n Adding “1” to this offset number gives the absolute Binary Byte number in the ensemble where Data Type #1 begins (the first byte of the ensemble is Binary Byte #1).
9–10	Offset for data type #2	
11–12	Offset for data type #3	
…	…	Sequence continues for up to n data types
(2n+5)-(2n+6)/2	Offset for data type #n	

Fixed Leader Data Format

Byte#	Field name	Notes
1–2/2	FIXED LEADER ID	Stores the Fixed Leader identification word (00 00h).
3/1	CPU F/W VER.	Contains the version number of the CPU firmware.
4/1	CPU F/W REV.	Contains the revision number of the CPU firmware.

Byte#	Field name	Notes

5–6/2	SYSTEM CONFIGURATION	This field defines the Navigator hardware configuration. Convert this field (2 bytes, LSB first) to binary and interpret as follows:
		
		LSB (Byte 5):
		`BITS 76543210`
		`-----000 75-kHz SYSTEM`
		`-----001 150-kHz SYSTEM`
		`-----010 300-kHz SYSTEM`
		`-----011 600-kHz SYSTEM`
		`-----100 1200-kHz SYSTEM`
		`-----101 2400-kHz SYSTEM`
		`----0--- CONCAVE BEAM PAT.`
		`----1--- CONVEX BEAM PAT.`
		`--00---- SENSOR CONFIG #1`
		`--01---- SENSOR CONFIG #2`
		`--10---- SENSOR CONFIG #3`
		`-0------ XDCR HD NOT ATT.`
		`-1------ XDCR HD ATTACHED`
		`0------- DOWN FACING BEAM`
		`1------- UP-FACING BEAM`
		
		MSB (Byte 6):
		`BITS 76543210`
		`------00 15E BEAM ANGLE`
		`------01 20E BEAM ANGLE`
		`------10 30E BEAM ANGLE`
		`------11 OTHER BEAM ANGLE`
		`0100---- 4-BEAM JANUS CONFIG`
		`0101---- 5-BM JANUS CFIG DEMOD)`
		`1111---- 5-BM JANUS CFIG.(2 DEMOD)`
		Example: Hex 5249 (e.g., hex 49 followed by hex 52) identifies a 150-kHz system, convex beam pattern, down-facing, 30E beam angle, 5 beams (3 demods).

7/1	REAL/SIM FLAG	This field is set by default as real data (0).
8/1	LAG LENGTH	Lag Length. The lag is the time period between sound pulses. This is varied, and therefore of interest in, at a minimum, for the WM5, WM8 and WM11 and BM7 commands.
9/1	NUMBER OF BEAMS	Contains the number of beams used to calculate velocity data (not physical beams). The Navigator needs only three beams to calculate water-current velocities. The fourth beam provides an error velocity that determines data validity. If only three beams are available, the Navigator does not make this validity check.
10/1	NUMBER OF CELLS {WN}	Contains the number of depth cells over which the Navigator collects data (WN-command). Scaling: LSD = 1 depth cell; Range = 1 to 128 depth cells

Byte#	Field name	Notes
11–12/2	PINGS PER ENSEMBLE {WP}	Contains the number of pings averaged together during a data ensemble (WP-command). If WP = 0, the Navigator does not collect the WD water-profile data. Note: The Navigator automatically extends the ensemble interval (TE) if the product of WP and time per ping (TP) is greater than TE (e.g., if WP x TP > TE). Scaling: LSD = 1 ping; Range = 0 to 16,384 pings
13–14/2	DEPTH CELL LENGTH {WS}	Contains the length of one depth cell (WS-command). Scaling: LSD = 1 centimeter; Range = 1 to 6400 cm (210 feet)
15–16/2	BLANK AFTER TRANSMIT {WF}	Contains the blanking distance used by the Navigator to allow the transmit circuits time to recover before the receive cycle begins (WF-command). Scaling: LSD = 1 centimeter; Range = 0 to 9999 cm (328 feet)
17/1	PROFILING MODE {WM}	Contains the Signal Processing Mode. This field will always be set to 1.
18/1	LOW CORR THRESH {WC}	Contains the minimum threshold of correlation that water profile data can have to be considered good data (WCcommand). Scaling: LSD = 1 count; Range = 0 to 255 counts
19/1	NO. CODE REPS	Contains the number of code repetitions in the transmit pulse. Scaling: LSD = 1 count; Range = 0 to 255 counts
20/1	%GD MINIMUM {WG}	Contains the minimum percentage of water-profiling pings in an ensemble that must be considered good to output velocity data. Scaling: LSD = 1 percent; Range = 1 to 100 percent
21–22/2	ERROR VELOCITY MAXIMUM {WE}	This field, initially set by the WE-command, contains the actual threshold value used to flag water-current data as good or bad. If the error velocity value exceeds this threshold, the Navigator flags all four beams of the affected bin as bad. Scaling: LSD = 1 mm/s; Range = 0 to 5000 mm/s
23/1	TPP MINUTES	These fields, set by the TP-command, contain the amount of time between ping groups in the ensemble. Note: The Navigator automatically extends the ensemble interval (set by TE) if (WP x TP > TE).
24/1	TPP SECONDS	
25/1	TPP HUNDREDTHS {TP}	

26/1	COORDINATE TRANSFORM {EX}	Contains the coordinate transformation processing parameters (EX-command). These firmware switches indicate how the Navigator collected data.
		`xxx00xxx = NO TRANSFORMATION (BEAM COORDINATES)`
		`xxx01xxx = INSTRUMENT COORDINATES`
		`xxx10xxx = SHIP COORDINATES`
		`xxx11xxx = EARTH COORDINATES`
		`xxxxx1xx = TILTS (PITCH AND ROLL) USED IN SHIP OR EARTH TRANSFORMATION`
		`xxxxxx1x = 3-BEAM SOLUTION USED IF ONE BEAM IS BELOW THE CORRELATION THRESHOLD SET BY THE WC-COMMAND`
		`xxxxxxx1 = BIN MAPPING USED`

27–28/2	HEADING ALIGNMENT {EA} LSB	Contains a correction factor for physical heading misalignment (EA-command). Scaling: LSD = 0.01 degree; Range = -179.99 to 180.00 degrees

Byte#	Field name	Notes
29–30/2	HEADING BIAS {EB}	Contains a correction factor for electrical/magnetic heading bias (EB-command). Scaling: LSD = 0.01 degree; Range = -179.99 to 180.00 degrees

31/1	SENSOR SOURCE {EZ}	Contains the selected source of environmental sensor data (EZ-command). These firmware switches indicate the following.
		FIELD DESCRIPTION
		`x1xxxxxx = CALCULATES EC (SPEED OF SOUND) FROM ED, ES, AND ET`
		`xx1xxxxx = USES ED FROM DEPTH SENSOR`
		`xxx1xxxx = USES EH FROM TRANSDUCER HEADING SENSOR`
		`xxxx1xxx = USES EP FROM TRANSDUCER PITCH SENSOR`
		`xxxxx1xx = USES ER FROM TRANSDUCER ROLL SENSOR`
		`xxxxxx1x = USES ES (SALINITY) FROM CONDUCTIVITY SENSOR`
		`xxxxxxx1 = USES ET FROM TRANSDUCER TEMPERATURE SENSOR`
		Note: If the field = 0, or if the sensor is not available, the Navigator uses the manual command setting. If the field = 1, the Navigator uses the reading from the internal sensor or an external synchro sensor (only applicable to heading, roll, and pitch). Although you can enter a “2” in the EZ-command string, the Navigator only displays a 0 (manual) or 1 (int/ext sensor).

32/1	SENSORS AVAILABLE	This field reflects which sensors are available. The bit pattern is the same as listed for the EZ-command (above).
33–34/2	BIN 1 DISTANCE	This field contains the distance to the middle of the first depth cell (bin). This distance is a function of depth cell length (WS), the profiling mode (WM), the blank after transmit distance (WF), and speed of sound. Scaling: LSD = 1 centimeter; Range = 0 to 65535 cm (2150 feet)
35–36/2	XMIT PULSE LENGTH BASED ON {WT}	This field, set by the WT-command, contains the length of the transmit pulse. When the Navigator receives a <BREAK> signal, it sets the transmit pulse length as close as possible to the depth cell length (WS-command). This means the Navigator uses a WT command of zero. However, the WT field contains the actual length of the transmit pulse used. Scaling: LSD = 1 centimeter; Range = 0 to 65535 cm (2150 feet)
37–38/2	(starting cell) WP REF LAYER AVERAGE {WL} (ending cell)	Contains the starting depth cell (LSB, byte 37) and the ending depth cell (MSB, byte 38) used for water reference layer averaging (WL-command). Scaling: LSD = 1 depth cell; Range = 1 to 128 depth cells
39/1	FALSE TARGET THRESH {WA}	Contains the threshold value used to reject data received from a false target, usually fish (WA-command). Scaling: LSD = 1 count; Range = 0 to 255 counts (255 disables)
40/1	SPARE	Contains the CX-command setting. Range = 0 to 5
41–42/2	TRANSMIT LAG DISTANCE	This field, determined mainly by the setting of the WMcommand, contains the distance between pulse repetitions. Scaling: LSD = 1 centimeter; Range = 0 to 65535 centimeters
43–50/8	CPU BOARD SERIAL NUMBER	Contains the serial number of the CPU board.

Byte#	Field name	Notes
51–52/2	SYSTEM BANDWIDTH {WB}	Contains the WB-command setting. Range = 0 to 1
53/1	SPARE	Spare
54/1	BASE FREQUENCY INDEX	Base frequency index
55–59/4	SPARE	Spare

Variable Leader Data Format

Byte#	Field name	
1–2/2	VARIABLE LEADER ID 80h	Stores the Variable Leader identification word (80 00h).

3–4/2	ENSEMBLE NUMBER	This field contains the sequential number of the ensemble to which the data in the output buffer apply. Scaling: LSD = 1 ensemble; Range = 1 to 65,535 ensembles Note: The first ensemble collected is #1. At “rollover,” we have the following sequence:
		1 = ENSEMBLE NUMBER 1
		65535 = ENSEMBLE NUMBER 65,535 | ENSEMBLE
		0 = ENSEMBLE NUMBER 65,536 | #MSB FIELD
		1 = ENSEMBLE NUMBER 65,537 | (BYTE 12 INCR.)

5/1	RTC YEAR {TS}	These fields contain the time from the Navigator’s real-time clock (RTC) that the current data ensemble began. The TS-command (Set Real-Time Clock) initially sets the clock. The Navigator does account for leap years.
6/1	RTC MONTH {TS}	
7/1	RTC DAY {TS}	
8/1	RTC HOUR {TS}	
9/1	RTC MINUTE {TS}	
10/1	RTC SECOND {TS}	
11/1	RTC HUNDREDTHS {TS}	
12/1	ENSEMBLE # MSB	This field increments each time the Ensemble Number field (bytes 3,4) “rolls over.” This allows ensembles up to 16,777,215. See Ensemble Number field above.

13–14/2	BIT RESULT	This field contains the results of the Navigator’s Built-in Test function. A zero code indicates a successful BIT result.
		(BYTE 14 RESERVED FOR FUTURE USE)
		`BYTE 13   BYTE 14`
		`1xxxxxxx xxxxxxxx = RESERVED`
		`x1xxxxxx xxxxxxxx = RESERVED`
		`xx1xxxxx xxxxxxxx = RESERVED`
		`xxx1xxxx xxxxxxxx = DEMOD 1 ERROR`
		`xxxx1xxx xxxxxxxx = DEMOD 0 ERROR`
		`xxxxx1xx xxxxxxxx = RESERVED`
		`xxxxxx1x xxxxxxxx = TIMING CARD ERROR`
		`xxxxxxx1 xxxxxxxx = RESERVED`

Byte#	Field name	
15–16/2	SPEED OF SOUND {EC}	Contains either manual or calculated speed of sound information (EC-command). Scaling: LSD = 1 meter per second; Range = 1400 to 1600 m/s
17–18/2	DEPTH OF TRANSDUCER {ED}	Contains the depth of the transducer below the water surface (ED-command). This value may be a manual setting or a reading from a depth sensor. Scaling: LSD = 1 decimeter; Range = 1 to 9999 decimeters
19–20/2	HEADING {EH}	Contains the Navigator heading angle (EH-command). This value may be a manual setting or a reading from a heading sensor. Scaling: LSD = 0.01 degree; Range = 000.00 to 359.99 degrees
21–22/2	PITCH (TILT 1) {EP}	Contains the Navigator pitch angle (EP-command). This value may be a manual setting or a reading from a tilt sensor. Positive values mean that Beam #3 is spatially higher than Beam #4. Scaling: LSD = 0.01 degree; Range = -20.00 to +20.00 degrees
23–24/2	ROLL (TILT 2) {ER}	Contains the Navigator roll angle (ER-command). This value may be a manual setting or a reading from a tilt sensor. For up-facing Navigators, positive values mean that Beam #2 is spatially higher than Beam #1. For down-facing Navigators, positive values mean that Beam #1 is spatially higher than Beam #2. Scaling: LSD = 0.01 degree; Range = -20.00 to +20.00 degrees
25–26/2	SALINITY {ES}	Contains the salinity value of the water at the transducer head (ES-command). This value may be a manual setting or a reading from a conductivity sensor. Scaling: LSD = 1 part per thousand; Range = 0 to 40 ppt
27–28/2	TEMPERATURE {ET}	Contains the temperature of the water at the transducer head (ET-command). This value may be a manual setting or a reading from a temperature sensor. Scaling: LSD = 0.01 degree; Range = -5.00 to +40.00 degrees
29/1	MPT MINUTES	This field contains the Minimum Pre-Ping Wait Time between ping groups in the ensemble.
30/1	MPT SECONDS	
31/1	MPT HUNDREDTHS	
32/1	HDG STD DEV	These fields contain the standard deviation (accuracy) of the heading and tilt angles from the gyrocompass/ pendulums. Scaling (Heading): LSD = 1°; Range = 0 to 180° Scaling (Tilts): LSD = 0.1°; Range = 0.0 to 20.0°
33/1	PITCH STD DEV	
34/1	ROLL STD DEV	

35/1	ADC CHANNEL 0	These fields contain the outputs of the Analog-to-Digital Converter (ADC) located on the DSP board. The ADC sequentially samples one of the eight channels per ping group (the number of ping groups per ensemble is the maximum of the WP). These fields are zeroed at the beginning of the deployment and updated each ensemble at the rate of one channel per ping group. For example, if the ping group size is 5, then:
		`END OF ENSEMBLE No.     CHANNELS UPDATED`
		`      Start             All channels = 0`

36/1	ADC CHANNEL 1	
37/1	ADC CHANNEL 2	
38/1	ADC CHANNEL 3	
39/1	ADC CHANNEL 4	
40/1	ADC CHANNEL 5	
41/1	ADC CHANNEL 6	

Byte#	Field name	

42/1	ADC CHANNEL 7	`1          0, 1, 2, 3, 4`
		`2          5, 6, 7, 0, 1`
		`3          2, 3, 4, 5, 6`
		`4          7, 0, 8, 2, 3`
		`...        ...`
		Here is the description for each channel:
		`CHANNEL    DESCRIPTION`
		`0          XMIT CURRENT`
		`1          XMIT VOLTAGE`
		`2          AMBIENT TEMP`
		`3          PRESSURE (+)`
		`4          PRESSURE (-)`
		`5          ATTITUDE TEMP`
		`6          ATTITUDE`
		`7          CONTAMINATION SENSOR`
		Note that the ADC values may be “noisy” from sample to sample, but are useful for detecting long-term trends.

Byte#	Field name	

43-46/4	ERROR STATUS WORD (ESW) {CY?}	Contains the long word containing the bit flags for the CY? Command. The ESW is cleared (set to zero) between each ensemble. Note that each number above represents one bit set – they may occur in combinations. For example, if the long word value is 0000C000 (hexadecimal), then it indicates that both a cold wake-up (0004000) and an unknown wake-up (00008000) occurred.
		`Byte 43`
		`BITS  7 6 5 4 3 2 1 0`
		`x x x x x x x 1 Bus Error exception`
		`x x x x x x 1 x Address Error exception`
		`x x x x x 1 x x Illegal Instruction exception`
		`x x x x 1 x x x Zero Divide exception`
		`x x x 1 x x x x Emulator exception`
		`x x 1 x x x x x Unassigned exception`
		`x 1 x x x x x x Watchdog restart occurred`
		`1 x x x x x x x Battery Saver power`
		`Byte 44`
		`x x x x x x x 1 Pinging`
		`x x x x x x 1 x Not Used`
		`x x x x x 1 x x Not Used`
		`x x x x 1 x x x Not Used`
		`x x x 1 x x x x Not Used`
		`x x 1 x x x x x Not Used`
		`x 1 x x x x x x Cold Wakeup occurred`
		`1 x x x x x x x Unknown Wakeup occurred`
		`Byte 45`
		`x x x x x x x 1 Clock Read error occurred`
		`x x x x x x 1 x Unexpected alarm`
		`x x x x x 1 x x Clock jump forward`
		`x x x x 1 x x x Clock jump backward`
		`x x x 1 x x x x Not Used`
		`x x 1 x x x x x Not Used`
		`x 1 x x x x x x Not Used`
		`1 x x x x x x x Not Used`
		`Byte 46`
		`x x x x x x x 1 Not Used`
		`x x x x x x 1 x Not Used`
		`x x x x x 1 x x Not Used`
		`x x x x 1 x x x Power Fail (Unrecorded)`
		`x x x 1 x x x x Spurious level 4 intr (DSP)`
		`x x 1 x x x x x Spurious level 5 intr (UART)`
		`x 1 x x x x x x Spurious level 6 intr (CLOCK)`
		`1 x x x x x x x Level 7 interrupt occurred`

47–48/2	SPARE	Reserved for TRDI use.

Byte#	Field name	
49–52/4	PRESSURE	Contains the pressure of the water at the transducer head relative to one atmosphere (sea level). Output is in decapascals. Scaling: LSD=1 deca-pascal; Range=0 to 4,294,967,295 deca-pascals
53–56/4	PRESSURE SENSOR VARIANCE	Contains the variance (deviation about the mean) of the pressure sensor data. Output is in deca-pascals. Scaling: LSD=1 deca-pascal; Range=0 to 4,294,967,295 deca-pascals
57/1	SPARE	Spare
58/1	RTC CENTURY	These fields contain the time from the Navigator's Y2K compliant real-time clock (RTC) that the current data ensemble began. The TT-command (Set Real-Time Clock) initially sets the clock. The Navigator does account for leap years.
59/1	RTC YEAR	
60/1	RTC MONTH	
61/1	RTC DAY	
62/1	RTC HOUR	
63/1	RTC MINUTE	
64/1	RTC SECOND	
65/1	RTC HUNDREDTH	

Velocity Data Format

Byte#	Field name	Notes
1–2/2	Velocity ID	Stores the velocity data identification word (00 01h).
3–4/2	DEPTH CELL #1, VELOCITY 1	Stores velocity data for depth cell #1, velocity 1.
5–6/2	DEPTH CELL #1, VELOCITY 2	Stores velocity data for depth cell #1, velocity 2.
7–8/2	DEPTH CELL #1, VELOCITY 3	Stores velocity data for depth cell #1, velocity 3.
9–10/2	DEPTH CELL #1, VELOCITY 4	Stores velocity data for depth cell #1, velocity 4.
11–12/2	DEPTH CELL #2, VELOCITY 1	These fields store the velocity data for depth cells 2 through 128 (depending on the setting of the WN-command). These fields follow the same format as listed above for depth cell 1.
13–14/2	DEPTH CELL #2, VELOCITY 2	
15–16/2	DEPTH CELL #2, VELOCITY 3	
17–18/2	DEPTH CELL #2, VELOCITY 4	
…	↓ (SEQUENCE CONTINUES FOR UP TO 128 CELLS) ↓	
1019–1020/2	DEPTH CELL #128, VELOCITY 1	
1021–1022/2	DEPTH CELL #128, VELOCITY 2	
1023–1024/2	DEPTH CELL #128, VELOCITY 3	
1025–1026/2	DEPTH CELL #128, VELOCITY 4	

Correlation Magnitude, Echo Intensity and Percent-Good Data Format

Byte#	Field name	Notes
1–2/2	ID CODE	Stores the correlation magnitude/echo intensity/percent-good data identification word (00 02h/00 03h/00 04h).
3/1	DEPTH CELL #1, FIELD #1	Stores correlation magnitude/echo intensity/percent-good data for depth cell #1, beam #1.
4/1	DEPTH CELL #1, FIELD #2	Stores correlation magnitude/echo intensity/percent-good data for depth cell #1, beam #2.
5/1	DEPTH CELL #1, FIELD #3	Stores correlation magnitude/echo intensity/percent-good data for depth cell #1, beam #3.
6/1	DEPTH CELL #1, FIELD #4	Stores correlation magnitude/echo intensity/percent-good data for depth cell #1, beam #4.
7/1	DEPTH CELL #2, FIELD #1	These fields store correlation magnitude/echo intensity/percent-good data for depth cells 2 through 128 (depending on the WN-command) for all four beams. These fields follow the same format as listed above for depth cell 1.
8/1	DEPTH CELL #2, FIELD #2	
9/1	DEPTH CELL #2, FIELD #3	
10/1	DEPTH CELL #2, FIELD #4	
	↓ (SEQUENCE CONTINUES FOR UP TO 128 BINS) ↓	
511/1	DEPTH CELL #128, FIELD #1	
512/1	DEPTH CELL #128, FIELD #2	
513/1	DEPTH CELL #128, FIELD #3	
514/1	DEPTH CELL #128, FIELD #4	

Header Format

Byte#	Field name	Notes
1	Header ID	Stores the header identification byte (7Fh)
2	Data source ID	Stores the data source identification byte (7Fh for the Work-Horse ADCP)
3–4	Number of bytes in ensemble	This field contains the number of bytes from the start of the current ensemble up to, but not including, the 2-byte checksum
5	Spare	Undefined
6	Number of data types	This field contains the number of data types selected for collection. By default, fixed/variable leader, velocity, correlation magnitude, echo intensity, and percent good are selected for collection. This field will therefore have a value of six (4 data types + 2 for the Fixed/Variable Leader data).
7–8	Offset for data type #1	These field contains the internal memory address offset where the Navigator will store information for data type #n Adding “1” to this offset number gives the absolute Binary Byte number in the ensemble where Data Type #1 begins (the first byte of the ensemble is Binary Byte #1).
9–10	Offset for data type #2	
11–12	Offset for data type #3	
…	…	Sequence continues for up to n data types

Byte#	Field name	Notes
(2n+5)-(2n+6)/2	Offset for data type #n	

Bottom Track Data Format

Byte#	Field name	Notes
1–2/2	BOTTOM-TRACK ID	Stores the bottom-track data identification word (06 00h).
3–4/2	BT PINGS PER ENSEMBLE {BP}	Stores the number of bottom-track pings to average together in each ensemble (BP-command). If BP = 0, the ADCP does not collect bottom-track data. The ADCP automatically extends the ensemble interval (TE) if BP x TP > TE. Scaling: LSD = 1 ping; Range = 0 to 999 pings
5–6/2	BT DELAY BEFORE RE-ACQUIRE {BD}	Stores the number of ADCP ensembles to wait after losing the bottom before trying to reacquire it (BD-command). Scaling: LSD = 1 ensemble; Range = 0 to 999 ensembles
7/1	BT CORR MAG MIN {BC}	Stores the minimum correlation magnitude value (BCcommand). Scaling: LSD = 1 count; Range = 0 to 255 counts
8/1	BT EVAL AMP MIN {BA}	Stores the minimum evaluation amplitude value (BAcommand). Scaling: LSD = 1 count; Range = 1 to 255 counts
9/1	BT PERCENT GOOD MIN {BG}	Stores the minimum percentage of bottom-track pings in an ensemble that must be good to output velocity data (BGcommand).
10/1	BT MODE {BM}	Stores the bottom-tracking mode (BM-command).
11–12/2	BT ERR VEL MAX {BE}	Stores the error velocity maximum value (BE-command). Scaling: LSD = 1 mm/s; Range = 0 to 5000 mm/s (0 = did not screen data)
13–16/4	RESERVED	Reserved
17–18/2	BEAM#1 BT RANGE	Contains the two lower bytes of the vertical range from the ADCP to the sea bottom (or surface) as determined by each beam. This vertical range does not consider the effects of pitch and roll. When bottom detections are bad, BT Range = 0. See bytes 78 through 81 for MSB description and scaling. Scaling: LSD = 1 cm; Range = 0 to 65535 cm
19–20/2	BEAM#2 BT RANGE	
21–22/2	BEAM#3 BT RANGE	
23–24/2	BEAM#4 BT RANGE	
25–26/2	BEAM#1 BT VEL	The meaning of the velocity depends on the EX (coordinate system) command setting. The four velocities are as follows: a) Beam Coordinates: Beam 1, Beam 2, Beam 3, Beam 4 b) Instrument Coordinates: 1->2, 4->3, toward face, error c) Ship Coordinates: Starboard, Fwd, Upward, Error d) Earth Coordinates: East, North, Upward, Error
27–28/2	BEAM#2 BT VEL	
29–30/2	BEAM#3 BT VEL	
31–32/2	BEAM#4 BT VEL	
33/1	BEAM#1 BT CORR.	Contains the correlation magnitude in relation to the sea bottom (or surface) as determined by each beam. Bottom-track correlation magnitudes have the same format and scale factor as water-profiling magnitudes.
34/1	BEAM#2 BT CORR.	
35/1	BEAM#3 BT CORR.	
36/1	BEAM#4 BT CORR.	

Byte#	Field name	Notes
37/1	BEAM#1 EVAL AMP	Contains the evaluation amplitude of the matching filter used in determining the strength of the bottom echo. Scaling: LSD = 1 count; Range = 0 to 255 counts
38/1	BEAM#2 EVAL AMP	
39/1	BEAM#3 EVAL AMP	
40/1	BEAM#4 EVAL AMP	
41/1	BEAM#1 BT %GOOD	Contains bottom-track percent-good data for each beam, which indicate the reliability of bottom-track data. It is the percentage of bottom-track pings that have passed the ADCP's bottom-track validity algorithm during an ensemble. Scaling: LSD = 1 percent; Range = 0 to 100 percent
42/1	BEAM#2 BT %GOOD	
43/1	BEAM#3 BT %GOOD	
44/1	BEAM#4 BT %GOOD	
45–46/2	REF LAYER MIN {BL}	Stores the minimum layer size, the near boundary, and the far boundary of the BT water-reference layer (BL-command). Scaling (minimum layer size): LSD = 1 dm; Range = 0-999 dm Scaling (near/far boundaries): LSD = 1 dm; Range = 0-9999 dm
47–48/2	REF LAYER NEAR {BL}	
49–50/2	REF LAYER FAR {BL}	
51–52/2	BEAM#1 REF LAYER VEL	Contains velocity data for the water reference layer for each beam. Reference layer velocities have the same format and scale factor as water-profiling velocities (Table 34, page 139). The BL-command explains the water reference layer.
53–54/2	BEAM #2 REF LAYER VEL	
55–56/2	BEAM #3 REF LAYER VEL	
57–58/2	BEAM #4 REF LAYER VEL	
59/1	BM#1 REF CORR	Contains correlation magnitude data for the water reference layer for each beam. Reference layer correlation magnitudes have the same format and scale factor as water-profiling magnitudes (Table 5).
60/1	BM#2 REF CORR	
61/1	BM#3 REF CORR	
62/1	BM#4 REF CORR	
63/1	BM#1 REF INT	Contains echo intensity data for the reference layer for each beam. Reference layer intensities have the same format and scale factor as water-profiling intensities.
64/1	BM#2 REF INT	
65/1	BM#3 REF INT	
66/1	BM#4 REF INT	
67/1	BM#1 REF %GOOD	Contains percent-good data for the water reference layer for each beam. They indicate the reliability of reference layer data. It is the percentage of bottom-track pings that have passed a reference layer validity algorithm during an ensemble. Scaling: LSD = 1 percent; Range = 0 to 100 percent
68/1	BM#2 REF %GOOD	
69/1	BM#3 REF %GOOD	
70/1	BM#4 REF %GOOD	
71–72/2	BT MAX. DEPTH {BX}	Stores the maximum tracking depth value (BX-command). Scaling: LSD = 1 decimeter; Range = 80 to 9999 decimeters
73/1	BM#1 RSSI AMP	Contains the Receiver Signal Strength Indicator (RSSI) value in the center of the bottom echo as determined by each beam. Scaling: LSD ≈ 0.45 dB per count; Range = 0 to 255 counts
74/1	BM#2 RSSI AMP	
75/1	BM#3 RSSI AMP	
76/1	BM#4 RSSI AMP	
77/1	GAIN	Contains the Gain level for shallow water. See WJ-command.
78/1	(*SEE BYTE 17)	Contains the most significant byte of the vertical range from the ADCP to the sea bottom (or surface) as determined by
79/1	(*SEE BYTE 19)	

Byte#	Field name	Notes
80/1	(*SEE BYTE 21)	each beam. This vertical range does not consider the effects of pitch and roll. When bottom detections are bad, BT Range=0. See bytes 17 through 24 for LSB description and scaling. Scaling: LSD = 65,536 cm, Range = 65,536 to 16,777,215 cm
81/1	(*SEE BYTE 23)	
82–85/4	RESERVED	Reserved

8.2.9 Checksum Data Format

Byte#	Field name	Notes
1–2/2	CHECKSUM DATA	This field contains a modulo 65535 checksum. The Workhorse computes the checksum by summing all the bytes in the output buffer excluding the checksum.

8.2.10 $_ZDA Report

Description

This NMEA string outputs UTC,day,month,year and local time zone.

Format

$--ZDA,hhmmss.sss,xx,xx,xxxx,xx,xx*hh <cr><lf>

ZDA Formatting

Field	Description
$	Start character
--	Sender Code
ZDA	Header
hhmmss.sss	Hours, minutes, seconds, and decimal seconds
xx	Day, 0 to 31
xx	Month, 01 to 12
xxxx	Year
xx	Local Zone hours 00 to ±13
xx	Local Zone minutes 00 to 59
*hh	Terminator and checksum
<cr><lf>	return plus linefeed

Supported Input Format

$GPZDA,**162408.00**,02,04,2007,,*6C

MUST be UTC

8.2.11 $_GGA Report

Description

This NMEA string outputs longitude and latitude at a UTC time.

Format

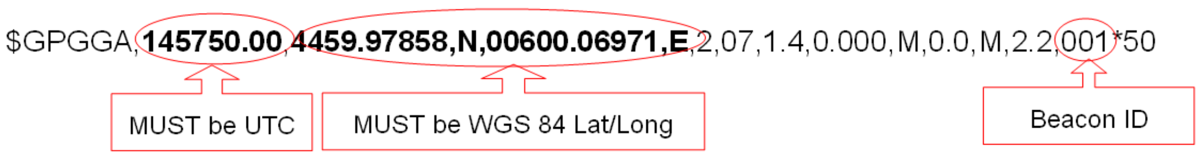
$--GGA, hhmmss.ss,llll.ll,a,yyyyy.yy,a,x,xx,x.x,x.xxx,M,x.x,M,x.x,xxxx*hh<cr><lf>

GGA Formatting

Field	Description
$	Start character
--	Sender Code (IN or GP)
GGA	Sentence Formatter
hhmmss.ss	UTC time
llll.ll,a	Latitude, N/S 2 fixed digits degrees, 2 fixed digits minutes, variable digits of decimal minutes.
yyyyy.yy,a	Longitude, E/W 3 fixed digits degrees, 2 fixed digits minutes, variable digits of decimal minutes.
x	GPS quality Indicator 0-8
xx	Number of satellites
x.x	Horizontal dilution of precision
x.xxx,M	Altitude above/below mean sea level (geoid), Metres
x.x,M	Geoidal Separation, Metres
x.x	Age of Differential data
xxxx	Differential Reference Station ID/SUSBL Beacon ID
*hh	Terminator and checksum
<cr><lf>	Terminator, return plus linefeed

Supported Input Formats

SUSBL example:



8.2.12 VTG

Description

This NMEA string outputs Speed and Course Over Ground (SOG & COG).

Format

$--VTG,x.x,T,x.x,M, x.x,N,x.x,K,a*hh<cr><lf>

Field	Description
$	Start character
--	Sender Code (e.g. GP, HE or IN)
VTG	Sentence Formatter
x.x,T	COG, degrees True
x.x,M,	COG, degrees Magnetic
x.x,N,	SOG, knots
x.x,K	SOG, km/hr
a	Mode Indicator
*hh	Terminator and checksum
<cr><lf>	Return plus linefeed

Example

$GPVTG,,T,,,,N,,K*03
$GPVTG,000.00,T,,M,0.00,N,0.0,K*60
$GPVTG,000.00,T,000.00,M,20.00,N,37.04,K*4C

Note

The Mode indicator can have any of the following values:
A = Autonomous
D = Differential
E = Estimated (dead reckoning)
M = Manual input
S = Simulator
N = Data not valid

8.2.13 PSONLOBS

This proprietary string is used for inputting LBL observations to Lodestar/SPRINT. It should be generated with an associated $PSONBCN message. These messages provide acoustic aiding to the INS algorithm.

Format

$PSONLOBS,tttttt.ttttt,ddd,ddd.ddd,dddd.ddd,dddd.ddd,dd.d,dd.d,d,c*hh<cr><lf>

Fields

Field	Description	Units
$PSONLOBS	Start character and format identifier	
tttttt.ttttt	INS system timestamp (Note 1)	Seconds
ddd	Beacon address	
ddd.ddd	Travel time (Note 2)	μSecond
dddd.ddd	Sound speed at beacon, as determined by acoustic system	m/s
dddd.ddd	Sound speed for range, as determined/used by acoustic system	m/s
dd.d	Signal to noise	dB
dd.d	Signal level (dB relative to full scale voltage [DBV-12])	dB
d	Cross correlation (Note 3)	
c	Status flag (A = accept, V = invalid)	
*hh	Terminator and checksum	
<cr><lf>	Return plus linefeed	

Example

$PSONLOBS,20138.186643,1706,444750.000,1485.000,1485.000,71.0,-2.0,89.0,A*7B

<cr><lf>

Notes

1. Time stamp is time of transceiver reception of the acoustic response signal (beginning). This can be expressed in the INS system time base, or in UTC. UTC timestamp is number of seconds since midnight with a negative sign to indicate UTC rather than Lodestar system time.

   e.g. 10:53:21.186643UTC should be sent as '-39201.186643'.

   The acoustic system can either determine INS system time from the INS augmented TCVR comms or get it via other means (in case TCVR comms is not augmented by INS).
2. Two way travel time including TAT.
3. Depending on the implementation the cross correlation may or may not be present.

   The $PSONBCN message will normally be expected before the $PSONLOBS message. If the $PSONBCN message is missing then the preceding message will be used. If the most recent PSONBCN is older than 180s then the observation is not used.

8.2.14 PSONLBLBCN

Description

This proprietary string contains the real world position of a calibrated (fixed) beacon. It is sent to Lodestar along with an associated $PSONLOBS message. These messages provide acoustic aiding to the INS algorithm.

Format

$PSONBCN,ttttttt.tttttt,dddd,d.ddddddd,d.ddddddd,d.ddd,d.ddd,ddddd,dd.d,d.d*hh<cr><lf>

Fields

Field	Description	Units
$PSONBCN	Start character and format identifier	
ttttttt.tttttt	INS timestamp (Note 1)	Seconds
dddd	Beacon address	
d.ddddddd	Latitude	Degrees
d.ddddddd	Longitude	Degrees
d.ddd	Depth	Metres
d.ddd	TAT (Turn Around Time)	ms
ddddd	Carrier frequency	Hz
dd.d	Horizontal error radius	Metres
d.d	Depth error	Metres
*hh	Terminator and checksum	
<cr><lf>	Return plus linefeed	

Example

$PSONBCN,922.672222,2306,28.2236437,-88.5303721,1693.373,200.000,25500,0.0,

0.0*59<cr><lf>

Notes

1. Timestamp can be omitted (empty) or set to zero. Time of arrival of message will be used in either case.
2. The Latitude and Longitude fields can contain up to 7 decimal places for a resolution of approximately 1 cm at the Equator.
3. The $PSONBCN message will normally be expected before the $PSONLOBS message. If the $PSONBCN message is missing then the preceding message will be used. If the most recent PSONBCN is older than 180s then the observation is not used.

8.2.15 Simrad SSB – SSBL Position Report ($PSIMSSB)

Reference: *Kongsberg APOS Release 4.2.2 Manual (29.April. 2005)*

Description

The PSIMSSB sentence contains the position of a SSBL beacon which is sent after each USBL measurement. The operator may define various parameters.

Format

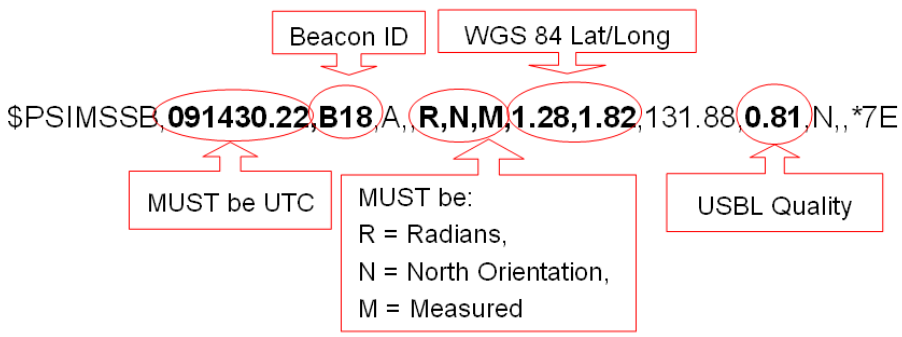
$PSIMSSB,hhmmss.ss,ccc,a,ccc,a,a,a,x.x,x.x,x.x,x.x,a,x.x,x.x*hh <cr><lf>

PSIMSSB Formatting

Field	Description
$	Start character
PSIM	Proprietary Simrad code
SSB	Sentence Formatter
hhmmss.ss	Empty or UTC time of reception
ccc	Beacon code, Examples: B01, B33, B47
a	Status, A for OK, V for not OK
ccc	Error_code, Empty or 3 character error code
a	C for Cartesian, P for polar, U for UTM coordinates, R for radians
a	Orientation, H vessel heading up; N for north; E for East:
a	Software filter, M=Measured, F=Filtered, P=Predicted
x.x	x coordinate
x.x	y coordinate
x.x	Depth in metres
x.x	Expected accuracy of the position
a	Additional info, N=None,C= Compass, I=Inclinometer, T=Time from Beacon to Transceiver
x.x	First add value, empty or Tp compass or Tp x inclination
x.x	Second add value, empty or Tp y inclination.
*hh	Terminator and checksum
<cr><lf>	return plus linefeed

Supported Input Format

Sonardyne Marksman/Ranger 2 and Kongsberg HiPAP:



8.2.16 $PSONSS Report

Description

Sound speed input from a non-specific source (depth not used)

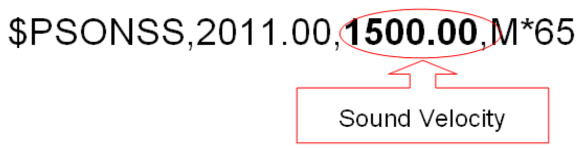
Format

$PSONSS,x.x,y.y,c*hh<cr><lf>

PSONSS Formatting

Field	Description
$	Start_character
PSONSS	Address
x.x	Depth
y.y	Sound Speed in Units per second
c	Units
*hh	Terminator and Checksum
<cr><lf>	Termination (0x0D 0x0A)

Supported Input Format



8.2.17 Valeport Sensor Telegram

Description

Data from the Valeport Mini SVS sensor. Assumes units are metres/second.

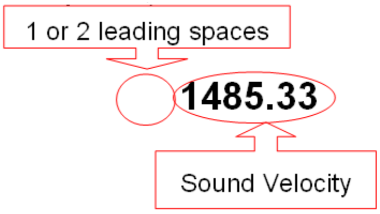
Format

<space>xxxx.xxx<cr><lf>

Valeport Sensor Telegram Formatting

Field	Description
<space>	A space character
xxxx.xxx	Sound Speed in metres per second
<cr><lf>	Termination (0x0D 0x0A)

Supported Input Format



8.2.18 XPOS

Description

External position and depth from a non-specific source.

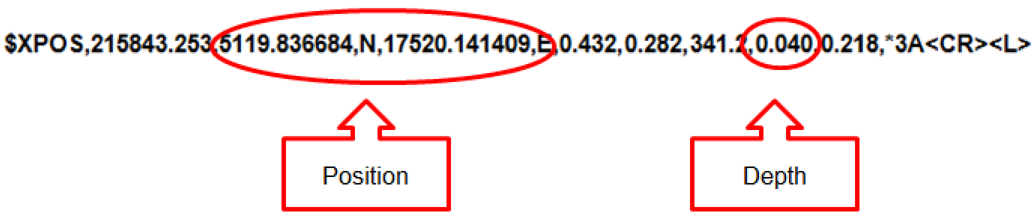
Format

$XPOS,hhmmss.sss,llll.llllll,a,yyyyy.yyyyyy,a,x.xxx,x.xxx,x.xxx,d.ddd,d.ddd,aa*hh<cr><lf>

XPOS Formatting

Field	Description	Units
$XPOS	Start Character	n/a
hhmmss.sss	UTC	
llll.llllll	Latitude: 2 fixed digits degrees, 2 fixed digits minutes, minimum 6 digits of decimal minutes (2mm resolution or better).	Degrees [0;90]
a	N/S	
yyyyy.yyyyyy	Longitude, 3 fixed digits degrees, 2 fixed digits minutes, minimum 6 digits of decimal minutes.	Degrees [0;180]
a	E/W	
x.xxx	Error ellipse, major axis. Can be empty	Metres
x.xxx	Error ellipse, minor axis. Can be empty	Metres
x.xxx	Error ellipse, orientation. Can be empty	Degrees
d.ddd	Depth. Can be empty	Metres
d.ddd	Depth standard deviation. Can be empty	Metres
aa	Spare*	n/a
*hh	Terminator and Checksum	
<cr><lf>	Carrier return and line feed	

Supported Input Format



8.3 LOG/OP Messages

8.3.1 CPU, UART & PWRSTAT

All three messages are ASCII based and provide a status of CPU loading, UART loading and Power Status at regular intervals.

8.3.2 Observation Status Message

Observation Status messages share a common generic header, with the remaining field being sensor dependent.

8.3.2.1 Generic Observation Header

Every message contains three (mostly) generic fields as detailed in *Table 8-3* which may be followed by further observation source specific fields.

Table 8-3 Generic Observation Status Fields

Byte#	Field name	Units	Data Type	Notes
1–6/6	timeTag	10^{-6} Seconds	Uint48	System time. Identical to the time tag of the associated (raw) observation data message. Unless otherwise specified, time tag is the time of arrival of the data at LS, e.g. time of the first message or packet byte.
7–8/2	reject		Bits x16	Rejection status bits; see *Table 8-4*
9–12/4	mahad		Float32	Mahalanobis distance. Indicates how well the observation matched the INS/Kalman expectation

The 4 MSB of the reject field are shown below. These bits are present in all observation status messages. The remaining 12 LSB are type specific as defined in the following sub-sections.

Table 8-4 Generic Rejection Bits

Bit	Name	Description
0 (LSB) 11	-	Type specific
12	misc	Miscellaneous: This bit is used to denote a problem not catered for by the other bits. It is set if the Kalman hasn't used this sensor data and none of the other rejection bits (0..11,13..15) are set. This may occur when the aiding data is not used by the Kalman filter as it has not yet initialised, but only if no other rejection bits are set. Another reason it could be set is if a general firmware error caused the observation to not be used.
13	ttag	Time tag issue: Any of the following: UTC time tag cannot be processed due to insufficient time sync Observation latency too large Observation time tag reasonability test failed
14	sig	Sigma test: Observation failed Kalman statistical testing. 0 otherwise. If bit 15 is set then SIG is 0 (test not performed).

Bit	Name	Description
15 (MSB)	dis	Disabled: Aiding source is disabled for AINS use (but still generating data). If this bit is set then all other bits may be set to 0 (not performed) or they may be set according to their otherwise specified functionality.

If the rejection field is all 0 then the observation was accepted.

8.3.2.2 OBSTZMD

This is generated once per Kalman cycle if and only if ZMD is enabled. Since ZMD is a “virtual” sensor it has no associated observation message and no message specific rejection bits. TimeTag is Kalman time of observation.

Table 8-5 OBSTZMD Specific Fields

Byte#	Field name	Units	Data Type	Notes
1–12/12	-	-	-	Generic; see *Table 8-3*
13–16/4	ResidualDepth	m	Float32	Residual, Depth

8.3.2.3 OBSTGPSPOS

The following table provides the type specific rejection bits which together with the generic bits described in *Table 8-4* combine to define the rejection bit field for the OBSTGPSPOS message.

Table 8-6 OBSTGPSPOS Specific Rejection Bits

Rejection Bit	Name	Description
0	-	=0
1	-	=0
2	-	=0
3	-	=0
4	QualNA	GPS accuracy metrics (e.g. NMEA GST) configured for use and not available
5	QI	NMEA GGA quality indicator unacceptable. Acceptable values are configurable see “GPS QUALITY” in command & control documentation.
6	-	=0
7	-	=0
8	-	=0
9	-	=0
10	-	=0
11	-	=0

The following table provides a full definition of the OBSTGPSPOS message.

Table 8-7 OBSTGPSPOS Specific Fields

Byte#	Field name	Units	Data Type	Notes
1–12/12	-	-	-	Generic; see *Table 8-3*
13–16/4	ResidualLat	rad	Float32	Residual, Latitude
17–20/4	ResidualLon	rad	Float32	Residual, Longitude
21–24/4	ResidualDepth	m	Float32	Residual, Depth. Zero if 2D
25–30/6	timeTag	10^{-6} Seconds	Uint48	System time of validity

8.3.2.4 OBSTSUSBL

The following table provides the type specific rejection bits which together with the generic bits described in *Table 8-4* combine to define the rejection bit field for the OBSTSUSBL message

Table 8-8 OBSTSUSBL Specific Rejection Bits

Rejection Bit	Name	Description
0	-	=0
1	-	=0
2	-	=0
3	ra	Reject Acoustic Acoustic observation status flag = ‘V’ (invalid). PSIMSSB.Coordinate_system setting not OK (should be ‘R’ – radians). PSIMSSB.SW_filter setting not OK (should be ‘M’ – measured). GGA.QI quality indicator unacceptable (=0). Note: Currently the FW checks GGA USBL messages against the GPS QUALITY setting – subject to change PPR 21558.
4	-	=0
5	-	=0
6	-	=0
7	-	=0
8	-	=0
9	lbi	Beacon info missing
10	-	=0
11	-	=0

The following table provides a full definition of the OBSTSUSBL message

Table 8-9 OBSTSUSBL specific fields

Byte#	Field name	Units	Data Type	Notes
1-12/12	-	-	-	Generic; see *Table 8-3*
13-14/2	Bcnid	-	Uint16	Beacon ID (address)
15/1	Obmsgtype	-	Uint8	Observation type: 0 – GGA 1 – $PSIMSSB
16-19/4	ResidualLat	Rad	Float32	Residual, latitude
20-23/4	ResidualLon	Rad	Float32	Residual, longitude
24-27/4	ResidualDepth	m	Float32	Residual, depth. Zero if 2D
28-33/6	timeTag	10^{-6} Seconds	Uint48	System time of validity

8.3.2.5 OBSTXPOS

The following table provides the type specific rejection bits which together with the generic bits described in *Table 8-4* combine to define the rejection bit field for the OBSTXPOS message.

Table 8-10 OBSTXPOS Specific Rejection Bits

Rejection Bit	Name	Description
0	-	=0
1	-	=0
2	-	=0
3	3DData	Depth Data not provided and INS XPOS USEVERTICAL set to 1
4	3DQual	Quality value unavailable (or unusable only when INS XPOS USEVERTICAL set to 0 AND Quality required – see “INS XPOS KFVPOS” in command and control documentation
5	qi	Quality indicator unacceptable. Value not provided when required see “INS XPOS KFHPOS” in command & control documentation.
6	-	=0
7	-	=0
8	-	=0
9	-	=0
10	-	=0
11	-	=0

The following table provides a full definition of the OBSTXPOS message.

Table 8-11 OBSTXPOS Specific Fields

Byte#	Field name	Units	Data Type	Notes
1–12/12	-	-	-	Generic; see *Table 8-3*
13/1	ObsSource	-	UInt8	Source of Observation: 0 = Command 1 = Message
14–17/4	ResidualLat	Rad	Float32	Residual, Latitude
18–21/4	ResidualLon	Rad	Float32	Residual, Longitude
22–25 /5	ResidualDepth	M	Float32	Residual, Depth. Zero if 2D.
26–31/6	timeTag	10^{-6} Seconds	Uint48	System time of validity

8.3.2.6 OBSTPDEPTH

The following table provides the type specific rejection bits which together with the generic bits described in *Table 8-4* combine to define the rejection bit field for the OBSTPDEPTH message.

Table 8-12 OBSTPDEPTH Specific Rejection Bits

Rejection Bit	Name	Description
0	-	=0
1	-	=0
2	-	=0
3	-	=0
4	-	=0
5	DevInv	Bathy (Winson) system devices flags not ok
6	qi	Quality indicator unacceptable. Value not provided when required see “INS PRESS NOISE” in command & control documentation. Should be provided when type is XDEPTH and PRESS NOISE is set to 0.0.
7	-	=0
8	-	=0
9	-	=0
10	-	=0
11	-	=0

The following table provides a full definition of the OBSTPDEPTH message.

Table 8-13 OBSTPDEPTH Specific Fields

Byte#	Field name	Units	Data Type	Notes
1–12/12	-	-	-	Generic; see *Table 8-3*
13/1	Obmsgtype	-	Uint8	0 – Keller 1 – PSONDEP 2 – DigiQuartz (m) 3 – DigiQuartz (psi) 4 – DigiQuartz (kPa) 5 – $__DPT 6 – PRDDIGIQO 7 – Winson 8 – Valeport SVX2 (DBAR) 9 XDEPTH
14–17/4	ResidualDepth	m	Float32	Residual, Depth
18–23/6	timeTag	10^{-6} Seconds	Uint48	System time of validity

8.3.2.7 OBSTSVS

The following table provides the type specific rejection bits which together with the generic bits described in *Table 8-4* combine to define the rejection bit field for the OBSTSVS message.

Table 8-14 OBSTSVS Specific Rejection Bits

Rejection Bit	Name	Description
0	-	=0
1	-	=0
2	-	=0
3	-	=0
4	-	=0
5	-	=0
6	unr	Measured sound speed unreasonable
7	-	=0
8	-	=0
9	-	=0
10	-	=0
11	-	=0

The following table provides a full definition of the OBSTSVS message.

Table 8-15 OBSTSVS Specific Fields

Byte#	Field name	Units	Data Type	Notes
1-12/12	-	-	-	Generic; see *Table 8-3*
13/1	Obmsgtype	-	Uint8	Observation type: 0 – VALEPORT 1 – $PSONSS 2 – Manual 3 – Auto
14-19/6	timeTag	10^{-6} Seconds	Uint48	System time of validity

8.3.2.8 OBSTDVL

The following table provides the type specific rejection bits which together with the generic bits described in *Table 8-4* combine to define the rejection bit field for the OBSTDVL message.

Table 8-16 OBSTDVL Specific Rejection Bits

Rejection Bit	Name	Description
0	-	=0
1	-	=0
2	svs	SVS bad (no good SVS observation in previous 60s)
3	cfg	Sensor configuration unsupported
4	evel	Velocity differences between beams too high
5	bsts	Bottom status bad
6	zbr	Zero beam range detected (no range measured on 1 or more beams)
7	zvel	Zero velocity measured (can be an outlier due to sensor error)
8	tout	Timeout – too long since last valid observation
9	velcu	Velocity change unreasonable (since previous valid observation)
10	-	=0
11	-	=0

The following table provides a full definition of the OBSTDVL message.

Table 8-17 OBSTDVL Specific Fields

Byte#	Field name	Units	Data Type	Notes
1–12/12	-	-	-	Generic; see *Table 8-3*
13–18/6	TOV	10^{-6} Seconds	Uint48	System time of validity
19/1	GroupID	-	Uint8	Group ID to associate observations that are batch processed together
20/1	Obmsgtype	-	Uint8	Bit 7 (MSB) = is beam Bit 6 = is water track Bits 0-5 = DVL message type: 0 = Generic PD4 1 = Generic PD5 2 = Generic PD0 8 = LinkQuest PD4 19 = Syrinx ASONDV
21–24/4	Sv	m/s	Float32	Sound speed used by Kalman filter
25–28/4	Residual1	m/s	Float32	Residual, Vx or Beam 1
29–32/4	Residual2	m/s	Float32	Residual, Vy or Beam 2
33–36/4	Residual3	m/s	Float32	Residual, Vz or Beam 3
37–40 /4	Residual4	m/s		Residual, Beam 4
/4	BeamReject	-	Uint32	Per Beam rejection status; see *Table 8-18*

Table 8-18 OBSTDVL Beam Rejection Status Fields

Rejection Bit(s)	Name	Equivalent Generic Specific Bit (see note 1)	Description
0–3	Sig1/2/3/4	sig	Sigma Test Beam X failed Kalman statistical testing
4–7	DvlRep1/2/3/4	misc	DVL Reported Beam X bad or disabled
8–11	Tout1/2/3/4	tout	Timeout – too long since last valid observation for Beam X
12–15	Velcu1/2/3/4	velcu	Beam X velocity change unreasonable (since previous valid observation for Beam X)
16–19	Xc1/2/3/4	misc	Cross correlation below C+C setting for Beam X
20–23	Snr1/2/3/4	misc	Signal to Noise ratio below C+C setting for Beam X
24–27	SigLev1/2/3/4	misc	Signal Level below C+C setting for beam X
28–31	BVelErr1/2/3/4	misc	Beam Velocity Error above limit set by C+C for Beam X

Notes on OBSTDVL when used for beam level aiding:

1. Bits in the Generic rejection bit field (made up of those bits specified in *Table 8-4* and *Table 8-16*) will only be used if they are correct for all four beams. If one or more beams are processed the Generic rejection bit field will be 0.

   If all four beams are rejected for different reasons the “misc” bit will be set

8.3.2.9 OBSTLBL

The following table provides the type specific rejection bits which together with the generic bits described in *Table 8-4* combine to define the rejection bit field for the OBSTLBL message.

For ObStLBL messages, the MISC bit will be set if the status flag in the associated PSONLOBS message is “V” (invalid).

Table 8-19 OBSTLBL Specific Rejection Bits

Rejection Bit	Name	Description
0	-	=0
1	-	=0
2	maxpred	Abs(observed range – predicted range) >= LS C&C limit
3	rrate	Range rate Observed range rate >= LS C&C limit
4	range	Range Below min range or larger than max range specified by LS C&C limit
5	tprevObs	Minimum previous observations Not enough previous observations in previous time specified by LS C&C limit
6	rsl	Reduced signal level Signal level filtered signal level <= -LS C&C limit
7	rsnr	Reduced signal to noise ratio SNR filtered SNR <= -LS C&C limit
8	sl	Signal Level Observed signal level <= LS C&C limit
9	snr	Signal to noise ratio Observed SNR <= LS C&C limit
10	lbi	Lvr or Bcn Info missing Transceiver lever arm or beacon configuration is undefined or, Age of PSONLBLLVR or PSONLBLBCN messages exceeded 180 seconds.
11	-	=0

The following table provide a full definition of the OBSTLBL message.

Table 8-20 OBSTLBL Specific Fields

Byte#	Field name	Units	Data Type	Notes
1–12/12	-	-	-	Generic; see *Table 8-3*
13–14/2	Bcnid	-	Uint16	Beacon ID (address)
15–18/4	Sv	m/s	Float32	Sound speed
19–22/4	ResidualRange	s	Float32	Residual, range
23–28/6	timeTag	10^{-6} Seconds	Uint48	System time of validity

8.3.2.10 OBSTZUPT

This is generated once per Kalman cycle if and only if ZUPT is enabled. Since ZUPT is a “virtual” sensor it has no associated observation message and no message specific rejection bits. TimeTag is Kalman time of observation.

Table 8-21 OBSTZUPT Specific Fields

Byte#	Field name	Units	Data Type	Notes
1–12/12	-	-	-	Generic; see *Table 8-3*
13–16/4	ResidualVx	m/s	Float32	Residual, Vel X
17–20/4	ResidualVy	m/s	Float32	Residual, Vel Y
21–24/4	ResidualVz	m/s	Float32	Residual, Vel Z

8.3.3 LSZDA

This message is produced by the Lodestar/SPRINT based on its own concept of UTC. The message format is as per the ZDA definition given in *Section 8.2.10*. For more details regarding how the the LSZDA is generated see *Section 7.11.6*.

8.3.4 PRDKELLBIN

Description

This message is produced whenever a new Pressure measurement is returned from a Keller pressure sensor, the time given in the message is the Time Of Validity of the pressure data contained within the message.

Format

Field#	Byte# (from 0)	Size (bytes)	Field name	Unit/LSB	Data type	Notes
1	0–5	6	timeTag	1e-6 sec	Uint48	Instrument time
2	7–10	4	Pressure		Float32	Converted from Bar reading given by sensor
3	11–12	2	Temperature	0.02DegC	Int16	

8.3.5 Time System Data (TMS)

The Time System data format is defined below.

Time System Data

Field#	Byte# (from 0)	Size (bytes)	Field name	Unit/LSB	Data type	Notes
1	0–5	6	sysTime	1e-6 sec	Uint48	Instrument time
2	6–13	8	utcTime	1e-6 sec	Uint64	UTC time – seconds since midnight 1970.01.01
3	14–19	6	timeSinceUpdate	1e-6 sec	Uint48	Time since last accepted UTC time update, e.g. from ZDA/PPS
4	20–23	4	stdDev	sec	Float32	Expected standard deviation of UTC time field
5	24	1	Source	logical	Uint8	Currently used source of UTC sync
6	25	1	ppsRising	logical	Uint8	0: PPS valid on falling edge (low to high voltage). 1: PPS valid on rising edge.
7	26	1	zdaCount	#	Uint8	LS byte of ZDA message count
8	27	1	ppsCount	#	Uint8	LS byte of PPS message count
9	28	1	zdaRejCount	#	Uint8	LS byte of ZDA message rejection count
10	29	1	ppsRejCount	#	Uint8	LS byte of PPS signal rejection count
11	30	1	ppsZdaProcCount	#	Uint8	LS byte of accepted PPS/ZDA pairs
12	31	1	filtResetCount	#	Uint8	LS byte of UTC filter reset count

Note

Source of RTC to UTC update; 0 = NO SOURCE; 1 = SPRINT UNIT RTC; 2 = Standalone GPZDA; 3 = Standalone GPGGA; 4 = GPZDA 1PPS.

Example: Converting IMU time tag from [sys] to [utc]

imu.timeTag[sys] = 1234567890 usec

Get the preceding time system message:

timeSys.sysTime = 1234101010 usec

timeSys.utcTime = 1254273030984001 usec

timeSys.stdDev = 0.0000124 sec   low std.dev. => UTC can be trusted

imu.timeTag [utc] = imu.timeTag[sys] + (timeSys.utcTime timeSys.sysTime)

= 1234567890 + (1254273030984001 1234101010) usec

= 1254273031450881 usec = 20090930T011031 (ISO 8601) = 2009-09-30 01:10:31

8.3.6 BIST

Description

The Built-In Self Test (BIST) message provides a summary of errors detected within a BIST cycle (2 seconds) and additional LS state information. Unless explicitly stated, an error indication is NOT persisted across cycles if the underlying cause has cleared or error event has not re-occurred.

BIST is generic, supporting all AHRS/INS applications. Each application will use only the relevant subset of the information provided as defined in the application specific ICD. Where possible, BIST information is accumulated into high level "go/no go" bits, thus simplifying the interface/use.

Format

The BIST message is logically subdivided into five blocks as shown below:

Table 8-22 BIST Message Blocks

Byte#	Field name	Unit(LSB)	Data type	Note
1–6/6	Time Tag	10^{-6} sec	Uint48	System time
7–14/8	Firmware Version		Uint64	FW Version: Major, Minor, Interim, Build. 16 bits each, Build number and LSB first.
15–22/8	IMU		Bits (x64)	
23–30/8	Comms		Bits (x64)	
31–38/8	CCA		Bits (x64)	Circuit Card Assembly
39–40/2	AHRS		Bits (x16)	
41–48/8	AINS		Bits (x64)	

Each block is described in the following tables.

Table 8-23 BIST – IMU Block

IMU bit#	Field name	Note/bit set
0	bIMUNoGo	Serious hardware issue (no go) (Note 1)
1	bISANotOk	ISA/IMU data not ok
2	bFWNotStarted	Firmware not started
3	bGyroPwrNotOk	Gyro power not ok
4	bXGyroProblem	X gyro problem – tripped or sensor status not 'Normal'
5	bYGyroProblem	Y gyro problem – tripped or sensor status not 'Normal'
6	bZGyroProblem	Z gyro problem – tripped or sensor status not 'Normal'
7	bExtPwrNotOk	External power supply not ok
8	bBatteryNotOk	Battery not ok
9	bRTCNotOk	Real Time Clock not ok
10	bXAccelSensorTempNotOk	X accelerometer sensor temperature not ok (> 80 ºC)

IMU bit#	Field name	Note/bit set
11	bYAccelSensorTempNotOk	Y accelerometer sensor temperature not ok (> 80 ºC)
12	bZAccelSensorTempNotOk	Z accelerometer sensor temperature not ok (> 80 ºC)
13	bXAccelBoardTempNotOk	X accelerometer board temperature not ok (> 80 ºC)
14	bYAccelBoardTempNotOk	Y accelerometer board temperature not ok (> 80 ºC)
15	bZAccelBoardTempNotOk	Z accelerometer board temperature not ok (> 80 ºC)
16	bAccelRangeNotOk	Accelerometer out of range
17	bAHRSResultNotOk	AHRS result not ok (Note 2)
18	bISATestModeEnabled	Not Used
19–31		Spare
32	bShutdownReq	Shutdown requested
33	bCurrentFlashNotUsed	Current flash not used
34	bPICNotAuth	PIC not authenticated
35	bOutputRestricted	Outputs Restricted for Export Reasons
36–63		Spare

Notes

1. "IMU no go" is a high level bit indicating serious HW error causing all AHRS/INS applications to fail or be unreliable. Bit is cumulative (or) result of bits 1-6,10-12, 16-17, 32-34 and (bit7 & bit8), e.g. "no go" is set if both 7 and 8 are set.
2. The "AHRS result not ok" bit is set if any of bits 1, 4-6, 10-12, 16 are set. When these bits become low, bit 17 remains latched high until a GC RST followed by an INS RST is done. None of these bits must be high between the GC RST and the INS RST otherwise bit 17 remains high.

Table 8-24 BIST – Comms Block

Comms bit#	Field name	Note/bit set
0	bUART0Rx	UART0 rx
1	bUART1Rx	UART1 rx
2	bUART2Rx	UART2 rx
3	bUART3Rx	UART3 rx
4	bUART4Rx	UART4 rx
5	bUART0Tx	UART0 tx
6	bUART1Tx	UART1 tx
7	bUART2Tx	UART2 tx
8	bUART3Tx	UART3 tx
9	bUART4Tx	UART4 tx
10	bTRIG1Rx	TRIG1 rx
11	bTRIG2Rx	TRIG2 rx
12	bTRIG3Rx	TRIG3 rx

Comms bit#	Field name	Note/bit set
13	bTRIG4Rx	TRIG4 rx
14	bUART0GeneralError	UART0 Rx General error (timetag overflow or frame or parity error)
15	bUART1GeneralError	UART1 Rx General error (timetag overflow or frame or parity error)
16	bUART2GeneralError	UART2 Rx General error (timetag overflow or frame or parity error)
17	bUART3GeneralError	UART3 Rx General error (timetag overflow or frame or parity error)
18	bUART4GeneralError	UART4 Rx General error (timetag overflow or frame or parity error)
19	bUART0TxOverflowError	UART0 Tx Overflow error
20	bUART1TxOverflowError	UART1 Tx Overflow error
21	bUART2TxOverflowError	UART2 Tx Overflow error
22	bUART3TxOverflowError	UART3 Tx Overflow error
23	bUART4TxOverflowError	UART4 Tx Overflow error
24	bTRIG1_RxOverrun	TRIG1 Rx Overrun
25	bTRIG2_RxOverrun	TRIG2 Rx Overrun
26	bTRIG3_RxOverrun	TRIG3 Rx Overrun
27	bTRIG4_RxOverrun	TRIG4 Rx Overrun
28	bSD	SD Rx
29	bETHERNET0_Rx	Ethernet Port0 rx
30	bETHERNET0_Tx	Ethernet Port0 tx
31	bETHERNET1_Rx	Ethernet Port1 rx
32	bETHERNET1_Tx	Ethernet Port1 tx
33	bETHERNET2_Rx	Ethernet Port2 rx
34	bETHERNET2_Tx	Ethernet Port2 tx
35	bETHERNET3_Rx	Ethernet Port3 rx
36	bETHERNET3_Tx	Ethernet Port3 tx
37	bETHERNET4_Rx	Ethernet Port4 rx
38	bETHERNET4_Tx	Ethernet Port4 tx
39	bETHERNET0_Error	Ethernet Port0 Error
40	bETHERNET1_Error	Ethernet Port1 Error
41	bETHERNET2_Error	Ethernet Port2 Error
42	bETHERNET3_Error	Ethernet Port3 Error
43	bETHERNET4_Error	Ethernet Port4 Error
44	bPTP_Rx	Not used
45	bPTP_Tx	Not used

Comms bit#	Field name	Note/bit set
46		
47	bETHERNET_Reset	Ethernet Reset
48	bMuxChecksumError	Multiplex Protocol Checksum error
49	bUART0RxParityError	UART 0 Rx Parity Error
50	bUART1RxParityError	UART 1 Rx Parity Error
51	bUART2RxParityError	UART 2 Rx Parity Error
52	bUART3RxParityError	UART 3 Rx Parity Error
53	bUART4RxParityError	UART 4 Rx Parity Error
54	bUART0RxOverrunError	UART 0 Rx Overrun Error
55	bUART1RxOverrunError	UART 1 Rx Overrun Error
56	bUART2RxOverrunError	UART 2 Rx Overrun Error
57	bUART3RxOverrunError	UART 3 Rx Overrun Error
58	bUART4RxOverrunError	UART 4 Rx Overrun Error
59	bUART0RxFrameError	UART 0 Rx Frame Error
60	bUART1RxFrameError	UART 1 Rx Frame Error
61	bUART2RxFrameError	UART 2 Rx Frame Error
62	bUART3RxFrameError	UART 3 Rx Frame Error
63	bUART4RxFrameError	UART 4 Rx Frame Error

Note

The Ethernet index entries are in increasing port number order; refer to the SETTINGS & CMDS msgs to determine which Ethernet port is for which BIST entry.

Table 8-25 BIST – CCA block

CCA bit#	Field name	Note/bit set
0	bCurrentFlashUsed	Current Flash used
1	bOldFlashUsed	Old Flash used
2	bCurrentFactoryUsed	Current Factory used
3	bOldFactoryUsed	Old Factory used
4	bPICUserFactory	PIC Factory user
5	bPICUserEngineer	PIC Engineer user
6	bPICUserCustomer	PIC Customer user
7	bPICIMUBitSet	PIC App – IMU bit set
8	bPICAppAll	PIC App – All
9	bPICAppAHRSOnly	PIC App – AHRS only
10	bPICAppOptUSBL	PIC App – optimised USBL
11	bPICAppDPINS	PIC App – DPINS

CCA bit#	Field name	Note/bit set
12	bPICAppSubsea	PIC App – Subsea
13	bPICAppGPSINS	PIC App – GPSINS
14		Spare
15	bUARTStartupNotOk	Problem setting up UARTs at startup
16	bSDNotInit	SD card not initialised
17	bSDDeleteNOTOK	SD card file or directory deletion not ok
18	bT1PwrPassOn	Power Pass on T1 On
19	bT2PwrPassOn	Power Pass on T2 On
20	bT1PwrPassTripped	Power Pass on T1 Tripped
21	bT2PwrPassTripped	Power Pass on T2 Tripped
22	Not Used	
23	bSDRW	Problem reading from or writing to the SD card
24	bPICAppRTIMU	PIC App – Real Time IMU Output bit set
25	bPICHWIssue	PIC HW problem
26	bC1PwrPassOn	Power Pass on C1 On
27	bC1PwrPassTripped	Power Pass on C1 Tripped
28	bDVLTrigNoise	Noise detected on Trigger output configured for DVL
29	bDVLTrigShort	Output DVL trigger not detected, output line may be shorted
30	bLSPPSTrigNoise	Noise detected on Trigger output configured for Lodestar generated PPS
31	bLSPPSTrigShort	Output of Lodestar PPS not detected, output line may be shorted
32	bFWIssue	Firmware issue
33–63		Spare

Table 8-26 BIST – AHRS block

AHRS bit#	Field name	Note/bit set
0		Reserved
1	bGCNotSettled	Unsettled
2	bGCNotPosAided	Not position aided for > 10s
3	bGCNotVelAided	Not velocity aided for > 10s
4–15		

Table 8-27 BIST – AINS block

AINS bit#	Field name	Note/bit set
0		Spare
1	bNotInit	AINS uninitialized

AINS bit#	Field name	Note/bit set
2	bNoOrient	No init orientation (AHRS not settled)
3	bNoVel	Not used
4	bNoPos	No init position available (less than x seconds old) for initialisation from enabled sensors (Note 1)
5	bNoDepth	No init depth available (less than x seconds old) for initialisation from enabled sensors (Note 1)
6-30		Spare
31	bZMDBLarge	ZMD Bias large (Note 2)
32	bPosLimit	Horizontal position 1DRMS high – INS auto re-initialize (Note 3)
33	bHdgInteg	Heading unreasonable (Note 4)
34	bAttInteg	Attitude unreasonable (Note 5)
35	bGyroBLarge	Gyro (bias) error large (Note 6)
36	bAccBLarge	Accel (bias) error large (Note 7)
37–63		Spare

Notes

1. x = Kalman delay + 1
2. ZMD Bias large when bias > (ZMD command MRms2 value) for > 120s.
3. Horizontal position 1DRMS high when 1DRMS > 1000 m (limit can be set by user) for more than 5 s.
4. Heading unreasonable when abs(INS hdg – AHRS hdg) > 2 degrees for more than 5 s.
5. Attitude unreasonable when abs (INS roll – AHRS roll) or abs (INS pitch – AHRS pitch) > 0.2 degrees for more than 5s.
6. Gyro bias large when bias > 3 sigma command value for > 120 s.
7. Accelerometer bias large when bias > 3 sigma command value for > 120 s.
8. Bits 32–63 = 0 when AINS is not initialised (bit 1 = 1).

8.3.7 TRG

Description

The purpose of this proprietary string is to record the 1 PPS input or any other trigger input or output such as DVLs and responders, and to log the trigger parameters.

Format

$PSONTRG,tttttttttttt,hhmmss.ssssss,d,c,c,tttttttt,tttttttt*hh <cr><lf>

Fields

Field	Description	Units
$PSONTRG	Start character and format identifier	
tttttttttttt	Trigger time in Lodestar System Time, seconds from start up, resolution to µs. 12 hex digits.	Seconds
hhmmss.ssssss	Trigger time converted from Lodestar Time System, seconds from start up, resolution to µs.	Seconds
d	Trigger port number (1 to 4)	
c	Trigger direction, (A = input, B = output)	
c	Sign of the trigger edge, “+” or “-“	
tttttttt	Width of the trigger pulse, resolution to µs. 8 hex digits.	Seconds
tttttttt	Period of the output pulse, resolution to µs. 8 hex digits.	Seconds
*hh	Terminator and checksum	
<cr><lf>	Return plus linefeed	

Example

$PSONTRG,00003FE06FAE,094020.500365,4,B,+,0000C350,000F4240*63<cr><lf>

Notes

1. Lodestar follows the Posix convention for expression of UTC time as a single number: The number of seconds elapsed since midnight 1970-01-01 not counting UTC leap seconds.
2. The width and period parameters are null fields if the trigger is an input.

8.3.8 NAVCAL

Description

This message is required for offline processing of INS data.

8.3.9 Navigation Quality Estimate (NavQual)

The navigation quality message reports the expected accuracy of the data given in the “nav” message.

Notes

1. Horizontal position 1DRMS = sqrt(posMajor^2 + posMinor^2).
2. CEP(50%) ~= 0.589 * (posMajor + posMinor).
3. Error ellipse (1 sigma) is 39.4% probability (e.g. 39.4% likelihood that true value is within ellipse).
4. 95% percent probability error ellipse is 2.447 * 1 sigma error ellipse.
5. Roll, pitch 1 sigma ~= max(stdLevN,stdLevE) for roll, pitch << 45deg.
6. Velocity rms = sqrt(velMajor^2+ velMinor^2).

Navigation Quality Estimate – Rate: 1 Hz

Byte#	Field name	Unit (LSB)/range	Data type	Note
0–5/6	timeTag	1e-6 sec	Uint48	System time
6–9/4	posMajor	m	Float32	Position (horizontal) error ellipse: - semi-major axis (1 sigma)
10–13/ 4	posMinor	-	Float32	- semi-minor axis (1 sigma)
14–17/4	dirPMajor	deg [0;360[	Float32	- direction of semi-major axis
18–21/4	stdDepth	m	Float32	Depth (1 sigma)
22–25/4	stdLevN	deg	Float32	Level error about North (1 sigma)
26–29/4	stdLevE	deg	Float32	Level error about East (1 sigma)
30–33/4	stdHeading	deg	Float32	Heading (1 sigma)
34–37/4	velMajor	m/s	Float32	Velocity (horizontal) error ellipse: semi-major axis (1 sigma)
38–41/4	velMinor	m/s	Float32	- semi-minor axis (1 sigma)
42–45/4	dirVMajor	deg [0;360[	Float32	- direction of semi-major axis
46–49/4	velDown	m/s	Float32	Down velocity (1 sigma)

8.3.10 Navigation Data (Nav)

The navigation (Nav) data message is the *generic* navigation output from SPRINT unit AINS and is closely related the navigation quality message (NavQual).

Note

Nav and NavQual are intended for advanced users including internal (Sonardyne) and external system integrators. Nav values are valid for vehicle CRP/frame, except acceleration which is valid for the IMU zero point but expressed in vehicle frame. For best accuracy it is recommended to use CRP=IMU zero point. AINS algorithm is the only source for the NAV message.

Table 8-28 Navigation Data

Byte#	Field name	Unit (LSB)	Data type	Note
0–5/6	timeTag	1e-6 sec	Uint48	System time
6–9/4	lat	2^-31*90 x° [-90;90]	Int32	+North (LSB ~= 0.5cm)
10–13/4	lon	2^-31*180° [-180; 180]	Int32	+East (LSB ~= 1cm @ Equator)
14–17/4	depth	1e-3 m	Int32	
18–19/2	altitude	1e-2 m	Uint16	Height above seabed (from DVL)
20–21/2	roll	2^-15*180° [-180; 180]	Int16	Angle between y and horizontal. Roll is positive when y is pointed below the horizontal (starboard down)
22–23/2	pitch	2^-15*180° [-90;90]	Int16	Angle between x and horizontal. Pitch is positive when x is pointed above the horizontal (bow up)

Byte#	Field name	Unit (LSB)	Data type	Note
24–25/2	heading	2^-15*180° [0;360]	Uint16	Angle between North and projection of X onto the horizontal (measured about down).
26–27/2	vx	1e-3 m/s	Int16	X-velocity (max ~32m/s)
28–29/2	vy	1e-3 m/s	Int16	Y-velocity (max ~32m/s)
30–31/2	vz	1e-3 m/s	Int16	Z-velocity (max ~32m/s)
32–33/2	wx	1e-2°/s	Int16	Angular rate about x axis (max ~327°/s)
34–35/2	wy	1e-2°/s	Int16	Angular rate about y axis (max ~327°/s)
36–37/2	wz	1e-2°/s	Int16	Angular rate about z axis (max ~327°/s)
38–39/2	ax	1e-3 m/s^2	Int16	X-acceleration (max ~ 3.2 G)
40–41/2	ay	1e-3 m/s^2	Int16	Y-acceleration (max ~ 3.2 G)
42–43/2	az	1e-3 m/s^2	Int16	Z-acceleration (max ~ 3.2 G)
44–45/2	mode	N/A	Bit16	Logical. Bit# 0: data valid 1:INS initialised 2: INS application not enable 3–14: Reserved 15: System failure

Note

Altitude = 0 imply invalid.

8.3.11 PRDID

Description

Pitch roll and heading message.

Format

$PRDID,PPP.PP,RRR.RR,hhh.hh*hh*hh<cr><lf>

Table 8-29 PRDID Formatting

Field	Description
$PRDID	Header
Pitch	Pitch, -30.0 to +30.0, degrees
Roll	Roll, -30.0 to +30.0, degrees
Heading	True Heading, 0 to 359.99, degrees
*hh	Terminator and checksum
<cr><lf>	Terminator, return plus linefeed

Example

$PRDID,-0.17,-0.59,172.66*77

Notes

1. The data string has variable length with leading zeros and minus signs added where necessary.
2. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.
3. The attitude measurements contained in the data string will be in real time.
4. There is no status indicator in the data string. This data string does include the optional checksum allowed within the NMEA 0183 standard.
5. The data string will include gyro heading information only if it is available. If there is no heading information available, the heading field will be null.

8.3.12 TSS1

Description

The TSS proprietary string outputs accelerations, heave, roll and pitch.

Format

:XXAAAASMHHHHQMRRRRSMPPPP<cr><lf>

Table 8-30 TSS1 Formatting

Field	Description
:	Start character
XX	Horizontal Acceleration (not populated by SPRINT Unit)
AAAA	Vertical Acceleration, vehicle frame
S	Space
M	Space if positive, minus if negative
HHHH	Heave
Q	Status Flag, H,h,F,f. Heading or Fully aided, settled or settling
M	Space if positive, minus if negative
RRRR	Roll
S	Space
M	Space if positive, minus if negative
PPPP	Pitch
<cr><lf>	Terminator, return plus linefeed

Example

:003D04 0000H-0058 -0017

Notes

1. Vertical acceleration is positive in the up direction.
2. Horizontal acceleration is not populated by the SPRINT Unit
3. The motion measurements contained in the data string will be in real time, valid for the instant when the system begins to transmit the string.
4. Motion measurements include ASCII-coded decimal values.
5. Heave measurements are in cm in the range –99.99 to +99.99 metres. Positive heave is above datum.
6. Roll and pitch measurements are in degrees in the range –99.99° to +99.99°. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.
7. Status flag H – The system is using heading from the settled gyrocompass.
8. Status flag h – The gyrocompass heading is not settled.
9. Status flag f – The system is receiving aiding data from both GGA and VTG NMEA messages but the gyrocompass is not settled.

10. Status flag F – The system is receiving aiding data from both GGA and VTG NMEA messages and the gyrocompass is settled.

8.3.13 TSS2

Description

his TSS proprietary string outputs heading, heave, roll and pitch.

Format

:DDDDDSMHHHHQMRRRRSMPPPPE<cr><lf>

Table 8-31 TSS2 Formatting

Field	Description
:	Start character
DDDDD	Heading x 100 degrees
S	Space
M	Space if positive, minus if negative
HHHH	Heave in centimetres
Q	Status Flag, H,h,F,f. Heading or Fully aided, settled or settling
M	Space if positive, minus if negative
RRRR	Roll x 100 degrees
S	Space
M	Space if positive, minus if negative
PPPP	Pitch x 100 degrees
E	Heading status flag, as for other TSS messages
<cr><lf>	Terminator, return plus linefeed

Example

:17263 0001H-0058 -0017A

Notes

1. The angle measurements are in hundredths (e.g. x 100).
2. The motion measurements contained in the data string will be in real time, valid for the instant when the System begins to transmit the string.
3. Motion measurements include ASCII-coded decimal values.
4. Heave measurements are in cm in the range –99.99 to +99.99 metres. Positive heave is above datum.
5. Roll and pitch measurements are in degrees in the range –99.99° to +99.99°. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.
6. Status flag is as for TSS1
   - Status flag H – The system is using heading from the settled gyrocompass.
   - Status flag h – The gyrocompass heading is not settled.

- Status flag f – The system is receiving aiding data from both GGA and VTG NMEA messages but the gyrocompass is not settled.
- Status flag F – The system is receiving aiding data from both GGA and VTG NMEA messages and the gyrocompass is settled.

7. Heading Status flag can take the following values:
   - A – If Status flag above is H or h
   - f – if Status flag above is f
   - F – if Status flag above is F

8.3.14 TSS3

Description

The TSS proprietary string outputs remote heave, heave, roll and pitch.

Format

:RMhhhhSMHHHHQMRRRRSMPPPP<cr><lf>

Table 8-32 TSS3 Formatting

Field	Description
:R	Start character and format identifier
M	Space or minus sign
hhhh	Remote Heave
S	Space
M	Space if positive, minus if negative
HHHH	Heave
Q	Status Flag, H,h,F,f. Heading or Fully aided, settled or settling
M	Space if positive, minus if negative
RRRR	Roll
S	Space
M	Space if positive, minus if negative
PPPP	Pitch
<cr><lf>	Terminator, return plus linefeed

Example

:R 0001 0001H-0059 -0017

Notes

1. After the start character (a colon, ASCII 3Ah) the TSS3 data string includes an upper case ‘R’ to identify the string as using TSS3 remote heave format.
2. The motion measurements contained in the data string will be in real time, valid for the instant when the System begins to transmit the string.
3. Motion measurements include ASCII-coded decimal values.

4. Heave measurements are in cm in the range –99.99 to +99.99 metres. Positive heave is above datum.
5. Roll and pitch measurements are in degrees in the range –99.99° to +99.99°. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.
6. Status flag H – The system is using heading from the settled gyrocompass.
7. Status flag h – The gyrocompass heading is not settled.
8. Status flag f – The system is receiving aiding data from both GGA and VTG NMEA messages but the gyrocompass is not settled.
9. Status flag F – The system is receiving aiding data from both GGA and VTG NMEA messages and the gyrocompass is settled.
10. Status flag A – General alarm.

8.3.15 EM1000

Description

Roll, pitch, heading and heave output suitable for use with Simrad EM series multibeam sonars.

Format

ABRRPPAAHH bytes 0–9

Table 8-33 EM1000 Formatting

Byte	Field		Field
0	A	MSB	Header, 0x00
1	BB	LSB	Header, 0x90
2	RR	LSB	Roll, Range +/- 20 deg. Units 0.01 deg.
3		MSB	
4	PP	LSB	Pitch, Range +/- 20 deg. Units 0.01 deg.
5		MSB	
6	AA	LSB	Heave +/- 20m, units 1 cm
7		MSB	
8	HH	LSB	Heading Range 0-359.99 deg. Units 0.01 deg.
9		MSB	

Example

00900200FF730000 hex

Notes

1. MSB = most significant byte, LSB = least significant byte
2. The data string is a 10-byte message of 16-bit 2's complement numbers, each expressed as two binary-coded digits.
3. Positive heave is above datum. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.
4. The motion measurements contained in the data string will be in real time.
5. The data string does not include a status flag.
6. The system updates the heading field in the data string only when it receives new heading information from the gyrocompass. Depending on the transmission rate of the gyrocompass there may therefore be a difference between the instantaneous heading and the value included in the data output string.
7. The gyro heading is NOT a 2's complement number.

8.3.16 EM3000

Description

Roll, pitch, heading and heave output suitable for use with Simrad EM3000 series multibeam sonars.

Format

ABRRPPAAHH bytes 0-9

Table 8-34 EM3000 Formatting

Byte	Field
A	Header, MSB, 0x00
B	Header LSB, 0x90 when settled or 0x91 when unsettled
RR	Roll, Range 0-359.99 deg. Units 0.01 deg.
PP	Pitch, Range 0-359.99 deg. Units 0.01 deg.
AA	Heave +/- 20m, units 1 cm
HH	Heading Range 0-359.99 deg. Units 0.01 deg.

Example

00900200FF730000 hex

Notes

1. MSB = most significant byte, LSB = least significant byte.
2. The data string is a 10-byte message of 16-bit 2’s complement numbers, each expressed as two binary-coded digits.
3. Positive heave is above datum. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.
4. The motion measurements contained in the data string will be in real time.
5. The Status byte = 91h for an unsettled unit or 90h for a settled unit.
6. The system updates the heading field in the data string only when it receives new heading information from the gyrocompass. Depending on the transmission rate of the gyrocompass there may therefore be a difference between the instantaneous heading and the value included in the data output string.
7. The gyrocompass heading is NOT a 2’s complement number.

8.3.17 PHTRO

Description

Pitch and roll message. This is similar to the NMEA 0183 standard. The units for the measurements are degrees; the angles are as described below.

Format

$PHTRO,x.xx,a,y.yy,b*hh<cr><lf>

Table 8-35 PHTRO Formatting

Field	Description
$PHTRO	Header
x.xx	x.xx is the pitch in degrees
,	comma
a	a is ‘M’ for bow up, ‘P’ for bow down
,	comma
y.yy	y.yy is the roll in degrees
,	comma
b	b is ‘B’ for port down, ‘T’ for port up
*hh	Terminator and checksum
<cr><lf>	Carriage return and linefeed characters

Example

$PHTRO,-0.17,P,-0.56,B*46

Notes

1. The data string has variable length with a leading zero if magnitude < 1 and minus signs added where necessary e.g. –0.59.
2. By default, positive roll is port-side up, starboard down, positive pitch is bow down, stern up. The “a” and “b” codes will be “P” and “T” respectively.
3. The attitude measurements contained in the data string will be in real time, valid for the instant when the system begins to transmit the first byte of the string.
4. There is no status indicator in the data string.

8.3.18 HDT

Description

NMEA True Heading.

Format

$HEHDT,x.x,T*hh<cr><lf>

Table 8-36 HDT Formatting

Field	Description
$	Start Character
HE	Talker identifier
HDT	Mnemonic for true heading
,	Comma separator
xxx.xxx	Heading in degrees and decimal fraction
,	Comma separator
T	Heading Type True/Grid/Magnetic
*hh	Checksum
<cr><lf>	Terminator, return plus linefeed

Example

$HEHDT,172.597,T*20

Note

The Heading type indicator is always ‘T’ when transmitted by the Lodestar, to indicate that heading information is with respect to true north.

8.3.19 THS

Description

This telegram is the NMEA defined “True heading and status” telegram actual vessel heading in degrees true produced by any device or system producing true heading.

Format

$__THS,XX.XX,a*hh<cr><lf>

Table 8-37 THS Formatting

Field	Description
$	Start Character
_	Talker identifier (HE)
THS	Mnemonic for true heading and status
,	Comma separator
XX.XX	Heading in degrees and decimal fraction
,	Comma separator
a	Mode indicator
*hh	Checksum
<cr><lf>	Terminator, return plus linefeed

Example

$HETHS,172.59,E*11

Notes

1. This sentence includes a “mode indicator” field providing critical safety related information about the heading data, and replaces the deprecated HDT sentence. The sender code for a north seeking gyrocompass is “HE”. For inertial navigation systems “IN” is used, though IN is for integrated navigation systems, see ISO 61162-1 for details.
2. Mode indicator states:
    - A = Autonomous (aided with GGA and VTG)
    - E = Estimated (dead reckoning, neither GGA and VTG are present)
    - M = Manual input
    - S = Simulator mode
3. V = Data not valid (gyrocompass not settled)

8.3.20 TEMP

Description

Internal temperature readings of Lodestar/SPRINT unit.

Format

$HETXT,d,d,dd,dd,dd,dd,dd,dd,dd*hh<cr><lf>

Table 8-38 TEMP Formatting

Field	Description
$	Start Character
HE	Talker identifier
TXT	Mnemonic for text message
d	Total number of messages (01 99)
d	Message number (01 – 99)
d	Text identifier (01 99)
dd.d	X sensor temperature, degrees Celsius
dd.d	Y sensor temperature, degrees Celsius
dd.d	Z sensor temperature, degrees Celsius
dd.d	X case temperature, degrees Celsius
dd.d	Y case temperature, degrees Celsius
dd.d	Z case temperature, degrees Celsius
*hh	checksum
<cr><lf>	Terminator, return plus linefeed

Example

$HETXT,1,1,66,43.1,43.5,42.9,43.8,42.9,43.9*7C

Notes

1. This message follows the standard spec for the __TXT message, where the first number is the total number of messages, the second number is the message number and the third the text identifier.
2. Following this there are 6 numbers and a checksum.

8.3.21 SON2

Description

Lodestar provides a Sonardyne proprietary SON2 telegram, consisting of UTC time, pitch, roll and heading with an estimated heading error. The heading error of the gyro-compass algorithm cannot be measured, only estimated from the sensor noise.

Format

:hhmmssssMRRRRRRMPPPPPPMHHHHHHMVVVS<cr><lf>

Table 8-39 SON2 Formatting

Field	Description
:	Start character
hhmmsssss	UTC time, hours, minutes, seconds and milliseconds
M	Space if positive, minus if negative
RRRRRR	Roll x 1000 degrees
M	Space if positive, minus if negative
PPPPPP	Pitch x 1000 degrees
M	Space if positive, minus if negative
HHHHHH	Heading x 1000 degrees
M	Space separator
VVV	Estimated variance
S	Status Flag, U,u,A,a,V,v,G,g
<cr><lf>	Terminator, return plus linefeed

Example

:152359000 000222-000022 359999 1234G

Notes

1. Positive roll is starboard down, port up.
2. Positive pitch is bow up, stern down.
3. The SON2 data string contains 39 characters in six data fields.
4. The time is UTC time expressed as time of day hours, minutes, seconds and milliseconds.
5. The angle measurements are in thousandths (e.g. x 1000 degrees)
6. The motion measurements contained in the data string will be in real time, valid for the instant when the System begins to transmit the string.
7. Due to the definition of the angles, the actual range of roll and pitch together are restricted. But the format allows for roll and pitch in degrees in the range –179.999° to +180.000°. Positive roll is port-side up, starboard down. Positive pitch is bow up, stern down.
8. The Status flag can take one of the following values:
   - If there is both VTG and GGA the status is A or a
   - If there is VTG only the status is V or v

- If there is GGA only the status is G or g
- If there is neither VTG nor GGA the status is U or u

Upper case denotes the Gyro has settled, lower case denotes the Gyro is settling.

If outputting u or U status, as soon as a VTG and/or GGA is received the status changes appropriately. However if VTG and/or GGA is not seen, it takes 5 seconds for the new (lesser) status to be updated on the message.

8.3.22 POSMV111

Description

This telegram contains data for delayed heave calculations along with time matched real-time heave data. Heave sign is positive down.

Table 8-40 POSMV111 Formatting

Byte	Field	Format	Description/value
1–4	Start	char	$GRP
5–6	ID	ushort	111
7–8	Byte count	ushort	76 (Bytes)
9–16	Time 1	double	seconds
17–24	Time 2	double	seconds
25–32	Distance tag	double	metres
33	Time type	byte	Bit2 set = Time1 UTC time (fixed) Bit4 set = Time2 POS time (fixed)
34	Distance type	byte	0 = N/A (fixed)
35–38	True heave	float	(,) metres
39–42	True heave RMS	float	[0,) metres
43–46	Status	Ulong	Bit0 set = True heave valid Bit1 set = realtime heave valid
47–50	Heave	float	(,) metres
51–54	Heave RMS	float	[0.) metres
55–62	Heave time 1	double	seconds
63–70	Heave time 2	double	seconds
71–74	Rejected IMU data count	ulong	[0,)
75–78	Out of range data count	ulong	[0,)
79–80	Pad	byte	0
81–82	Checksum	ushort	
83–84	Group end	char	$#

Notes

1. Time 1 is the system time of validity for the data. The type of data is indicated in the time type field. UTC time is the seconds of the week.

2. Time 2 is the system time of validity for the data. The type of data is indicated in the time type field. POS time is the time in seconds since power on.
3. The checksum is calculated so that the sum of short pairs (16 bits) over the complete telegram has a sum of zero.
4. Byte is 8 bits MSB first.
5. Short is the INTEL format 16 bits MSB first.
6. Long is 32 bits MSB first.
7. Float is INTEL format from IEEE-754 floating point definition.
8. Double is 8 bytes, MSB first.
9. MSB = Most Significant Bit, LSB = Least Significant Bit.

8.3.23 POSMV113

Description

This telegram contains quality data for delayed heave calculations.

Table 8-41 POSMV113 Formatting

Byte	Field	Format	Description/value
1–4	Start	char	$GRP
5–6	ID	ushort	113
7–8	Byte count	ushort	68 (Bytes)
9–16	Time 1	double	seconds
17–24	Time 2	double	seconds
25–32	Distance tag	double	metres
33	Time type	byte	Bit2 set = Time1 UTC time (fixed) Bit4 set = Time2 POS time (fixed)
34	Distance type	byte	0 = N/A (fixed)
35–42	Heave time 1	double	Seconds
43–50	Quality control 1	double	0 (fixed)
51–58	Quality control 2	double	0 (fixed)
59–66	Quality control 3	double	0 (fixed)
67–70	Status	Ulong	0 (fixed). Quality control not used.
71–72	Pad	byte	0
73–74	Checksum	ushort	
75–76	Group end	char	$#

Notes

1. Time 1 is the system time of validity for the data. The type of data is indicated in the time type field. UTC time is the seconds of the week.
2. Time 2 is the system time of validity for the data. The type of data is indicated in the time type field. POS time is the time in seconds since power on.
3. The checksum is calculated so that the sum of short pairs (16 bits) over the complete telegram has a sum of zero.
4. Byte is 8 bits MSB first.
5. Short is the INTEL format 16 bits MSB first.
6. Long is 32 bits MSB first.
7. Float is INTEL format from IEEE-754 floating point definition.
8. Double is 8 bytes, MSB first.
9. MSB = Most Significant Bit, LSB = Least Significant Bit.

8.3.24 PSONNAV

Description

The purpose of this proprietary message provides output navigation which consists of a UTC timestamp, position, depth, velocity, attitude and heading with associated accuracy estimates.

Format

$PSONNAV,hhmmss.sss,llll.llllll,a,yyyyy.yyyyyy,a,x.xxx,x.xxx,x.xxx,a,d.ddd,x.xxx,r.rrr,p.ppp,h.hhh,x.xxx,a,aaaaaaa, , , , , ,*hh<cr><lf>

Table 8-42 PSONNAV Formatting

Field	Description
$	Start character
PSONNAV	Address
hhmmss.sss	UTC Timestamp
llll.llllll,a	Latitude
yyyyy.yyyyyy,a	Longitude
x.xxx	Major Axis position error ellipse
x.xxx	Minor Axis position error ellipse
x.xxx	Direction of major Axis position error ellipse
a	Position Status (‘A’ – Valid, ‘V’ – Invalid)
d.ddd	Depth
x.xxx	Depth standard deviation
r.rrr	Roll
p.ppp	Pitch
h.hhh	heading
x.xxx	Heading standard deviation
a	Orientation status (‘A’ – Valid, ‘V’ – Invalid)
aaaaaaa	Sensor status (see Note 1)
, , , , , ,	Null fields for future use
<cr><lf>	return plus linefeed

Example Output Format

$PSONNAV,153239.443,5119.838453,N,00050.141452,W,0.155,0.155,1.861,A,-0.040,0.218,0.798,0.079,279.846,0.133,A,AI,,,,,,*46<cr><lf>

Note

The following table provides details as the possible character that can be used in the Sensor status field of the message.

Sensor/solution	Data received and some or all used within the last 1 second.	Data received within the last 1 second but none was used.
AHRS	‘A’	‘a’
LBL	‘B’	‘b’
Depth (pressure)	‘D’	‘d’
GNSS	‘G’	‘g’
AINS (Note 4)	‘I’	na
ZMD (“Zero Mean Depth”, e.g. surface ship)	‘M’	‘m’
DVL	‘V’	‘v’
USBL	‘U’	‘u’
SUSBL	‘S’	‘s’
XPOS	‘X’	‘x’
ZUPT	‘Z’	‘z’

8.3.25 LNAV & LNAVUTC

Description

The long navigation (LNav) and long navigation UTC (LNavUTC) data messages are two generic navigation outputs from Lodestar INS/AHRS that differ in only what time is represented in the timeTag field of the message. LNav contains System Time, whilst LNavUTC contains UTC. The remaining message content is essentially the combination of the Nav and NavQual messages.

The LNav/LNavUTC message can be sourced from either AHRS or INS. If the SRC is set to AHRS then the orientation fields will be populated with AHRS data, if the INS is also initialised the other message fields will also be populated. If the source selected is INS all fields are sourced from INS if the INS is initialised – if the INS is not initialised then data that is available from the AHRS algorithm will be used.

All data is transmitted LSB first.

Table 8-43 LNav Data

Byte# (from 1)	Size (bytes)	Field name	Units	Optional	Data type	Notes
1–6	6	Time Tag	10^{-6} seconds (LNAV) 10^{-5} seconds (LNAVUTC)	No	Uint48	System time, Note 12
7–10	4	Latitude	2^{-31}×90 deg	Yes, see Note 11	Int32	Latitude, Note 1
11–14	4	Longitude	2^{-31}×180 deg	Yes, see Note 11	Int32	Longitude, Note 2
15–18	4	Depth	10^{-3} metres	Yes, see Note 11	Int32	Depth below sea level, Note 3

Byte# (from 1)	Size (bytes)	Field name	Units	Optional	Data type	Notes
19–20	2	Altitude	10^{-2} metres	Yes, see Note 11	Uint16	Height above seabed, Note 4
21–22	2	Roll	2^{-15}×180 deg	No	Int16	Note 5
23–24	2	Pitch	2^{-15}×180 deg	No	Int16	Note 6
25–26	2	Heading	2^{-15}×180 deg	No	Uint16	Note 7
27–28	2	vN	10^{-3} m/s	Yes, see Note 11	Int16	Vehicle North-velocity
29–30	2	vE	10^{-3} m/s	Yes, see Note 11	Int16	Vehicle East-velocity
31–32	2	vDwn	10^{-3} m/s	Yes, see Note 11	Int16	Vehicle Down-velocity
33–34	2	wFwd	10^{-2} deg/s	No	Int16	Angular rate about Vehicle Fwd axis
35–36	2	wStbd	10^{-2} deg/s	No	Int16	Angular rate about Vehicle Stbd axis
37–38	2	wDwn	10^{-2} deg/s	No	Int16	Angular rate about Vehicle Dwn axis
39–40	2	aFwd	10^{-3} m/s^2	No	Int16	Vehicle Fwd-acceleration
41–42	2	aStbd	10^{-3} m/s^2	No	Int16	Vehicle Stbd-acceleration
43–44	2	aDwn	10^{-3} m/s^2	No	Int16	Vehicle Dwn-acceleration
45–48	4	posMajor	Metres	Yes, see Note 11	Float32	Horizontal position 1σ error ellipse (Note 8): - semi-major axis
49–52	4	posMinor	Metres	Yes, see Note 11	Float32	- semi-minor axis
53–56	4	dirPMajor	Degrees	Yes, see Note 11	Float32	- direction of semi-major axis
57–60	4	stdDepth	Metres	Yes, see Note 11	Float32	1σ depth error
61–64	4	stdLevN	Degrees	Yes, see Note 11	Float32	1σ level error about North (Note 9)
65–68	4	stdLevE	Degrees	Yes, see Note 11	Float32	1σ level error about East (Note 9)
69–72	4	stdHeading	Degrees	Yes, see Note 11	Float32	1σ heading error

Byte# (from 1)	Size (bytes)	Field name	Units	Optional	Data type	Notes
73–76	4	velMajor	m/s	Yes, see Note 11	Float32	Horizontal velocity 1σ error ellipse (Note 10): - semi-major axis
77–80	4	velMinor	m/s	Yes, see Note 11	Float32	- semi-minor axis
81–84	4	dirVMajor	Degrees	Yes, see Note 11	Float32	- direction of semi-major axis
85–88	4	velDown	m/s	Yes, see Note 11	Float32	1σ down velocity error
89–90	2	Status	N/A	No	Bit16	Note 13

Notes

1. Latitude, north is positive. 0.5 cm resolution.
2. Longitude, east is positive. 1 cm resolution at equator.
3. Depth, down is positive.
4. Height above seabed as measured by the DVL, contains the last valid altitude received from the DVL.
5. Roll is the angle between the Stbd-axis and horizontal. Roll is positive when Stbd is pointed below the horizontal (starboard down).
6. Pitch is the angle between the Fwd-axis and horizontal. Pitch is positive when Fwd is pointed above the horizontal (bow up).
7. Heading is the angle between North and projection of the Fwd-axis onto the horizontal (measured about the down direction).
8. Horizontal position 1σ error ellipse
   - Horizontal position 1DRMS = sqrt(posMajor^2 + posMinor^2)
   - CEP(50%) ~= 0.589 × (posMajor + posMinor)
   - Error ellipse (1σ) is 39.4% probability (e.g. 39.4% likelihood that true value is within ellipse)
   - 95% percent probability error ellipse is 2.447 × 1σ error ellipse
9. Roll & pitch 1σ ~= max(stdLevN,stdLevE) for roll, pitch << 45 deg.
10. Velocity RMS = sqrt(velMajor^2+ velMinor^2)
11. Will be populated with zero values if data unavailable (e.g. INS is not initialised/DVL data invalid)
12. For the LNAVUTC the time is the time since 1st Jan 1970 with a resolution of 0.01ms (not including leap seconds).
13. The status bits are described below (if bit = 0 then status is OK):

Table 8-44 LNav Status

Status Bit	Field name	Notes/Bit Set
0	bOrientationStatus	Orientation Invalid (e.g. AHRS not OK or unsettled)
1	bPosStatus	Position (& Velocity) Invalid (e.g. INS not OK or not initialised)
2	bAltitudeStatus	0 indicates that the altitude field has been updated in this message compared to the last time the LNAV message was sent. 1 indicates that the altitude data is either old (no update from DVL since last LNAV message sent) or invalid.
4	bOrientationSource	0 indicates Orientation source = AHRS, 1 indicates Orientation source = INS
5	bSubseaUSBLUsed	0 indicates data received and some or all used within the last second, otherwise 1
6	bDepthUsed	0 indicates data received and some or all used within the last second, otherwise 1
7	bDVLUsed	0 indicates data received and some or all used within the last second, otherwise 1
8	bLBLUsed	0 indicates data received and some or all used within the last second, otherwise 1
9	bZUPTUsed	0 indicates data received and some or all used within the last second, otherwise 1
10	bXPOSUsed	0 indicates data received and some or all used within the last second, otherwise 1
11	bGPSUsed	0 indicates data received and some or all used within the last second, otherwise 1
12	bZMDUsed	0 indicates data received and some or all used within the last second, otherwise 1
13	bUSBLUsed	0 indicates data received and some or all used within the last second, otherwise 1
14-15	Not Used	Reserved for future use

Section 9 – Command and Control

9.1 SYS Commands

Mnemonic	Purpose
LIST	Lists all parameters
CMDS LIST	Lists command parameters
NONDEFAULT LIST	Lists command parameters to be used to set up current functionality from Default Settings.
CMDS VERSION	Lists the commands version number
HELP	Helps with command syntax
RST	Reboot the system
EXIT	Exit the command line
SAVE	Saves a configuration
LOAD	Loads a configuration
NET	Sets or reads the IP address and mask
NET RST	Resets the Ethernet
NET PING	Pings an IP address
MAC	Reads the MAC address
LA	Sets or reads Remote Point Lever arms [2-7]
SHUTDOWN	Turns off the Lodestar
AUTOSHUTDOWN	Turns off the Lodestar if no external power for x secs.
SPRINTNAV	Reports whether the Lodestar/SPRINT is part of a combined unit (with Syrinx and pressure sensor)
BIST	Output the Lodestar BIST in ASCII format
CPU	Displays the CPU loading
UART	Displays the UART loading
PWRSTAT	Displays the Power Supplies and Battery Charge Status

9.2 GC Commands

Mnemonic	Purpose
LIST	Lists all GC parameters
HELP	Helps with command syntax
LAT	Sets or reads the latitude
SETTLE	Sets or reads the AHRS settling time
RST	Resets the AHRS

9.3 OP Commands

Mnemonic	Purpose
LIST	Lists all OP parameters
HELP	Helps with command syntax
SER	Sets the baud rate, number of data bits, the parity and the number of stop bits
BAUD	Sets the baud rate
DATA	Sets the number of data bits
PAR	Sets the parity
STOP	Sets the number of stop bits
PROT	Set signalling protocol (232, 485 …)
MSG	Lists the messages and their numbers or sets the messages to be output
ECHO	Enables or disables the echoing of user-entered characters
MULTIPLEX	Sets if port is to output in native mode or wrapped in binary format
HOLDOFF	Used for a network port as a timeout for data output
BREAK	Sends a serial break on a UART
NET	Sets up the protocol for a network port
CLOSE	Closes a network port

9.4 IN Commands

Mnemonic	Purpose
LIST	Lists all IN parameters
HELP	Helps with command syntax
MSG	Lists the messages and their numbers or sets the messages to be output
NET	Sets up the protocol for a network port message

9.5 PORT Commands

Mnemonic	Purpose
LIST	Lists all ISA parameters
HELP	Helps with command syntax
SNOOP	Sets or reads the snooping ports
PASS	Sets or reads the pass-through ports
PWRPASS	Sets or reads the power pass-through ports

9.6 INS Commands

Mnemonic	Purpose
LIST	Lists all INS parameters
HELP	Helps with command syntax
USE	Sets or reads the aiding messages for the AINS
KFHPOSRST	Sets or reads the INS Auto Reset value
KFHPOSBOOST	Sets or reads the value for boosting of horizontal position covariance
RELNAVOUT	Sets or reads the Relative Navigation settings values
GPS HDOPMAX	Sets or reads the maximum HDOP value
GPS USEVERTICAL	0=2D, 1=3D vertical aiding
SUSBL USEVERTICAL	0=2D, 1=3D vertical aiding
DVL KFSF	Sets or reads Scale factor and time constant. Bias/random noise, 1st order Markov RMS value.
DVL KFMA	Sets or reads RMS value and time constant. Boresight misalignment accuracy
XPOS	Reads the position. Sets the position at a time
XPOS USEVERTICAL	0=2D, 1=3D vertical aiding
XSAL	Sets or reads current salinity of sea water
XSV	Sets or reads current sound speed
RST	Resets the AINS

9.7 GPS Commands

Mnemonic	Purpose
LIST	Lists all GPS parameters
HELP	Helps with command syntax
LA	Sets or reads the GPS lever arms (m)
QUALITY	Sets or reads the min and max GPS quality accepted by the AINS.

9.8 SUSBL Commands

Mnemonic	Purpose
LIST	Lists all USBL parameters
HELP	Helps with command syntax
LA	Sets or reads the SUSBL lever arms (m)
TPDR	Sets or reads the transponder ID to be used from the PSIMSSB
TSOURCE	Sets or reads the Time SOURCE to be used for TOV
TSYNCSDLIM	Sets or reads the TSYS S.D. Limit to be used for rejection criteria

Mnemonic	Purpose
LATENCY	Sets or reads the Latency to be applied in the TOV calculations

9.9 LBL Commands

Mnemonic	Purpose
LIST	Lists all LBL parameters
HELP	Helps with command syntax
LA	Sets or reads the LBL lever arms (m)
SIGNAL	Sets or reads SNR/signal limits for outlier rejection
RANGE	Sets or reads range limits for outlier rejection
PASTOBSCNT	Sets or reads number of observations required
MAXTSINCEPASTTWT	Sets or reads time within which observations are expected

9.10 XPOS Commands

Mnemonic	Purpose
LIST	Lists all XPOS parameters
HELP	Helps with command syntax
LA	Sets or reads the XPOS lever arms (m)

9.11 ZMD Commands

Mnemonic	Purpose
LIST	Lists all ZMD parameters
HELP	Helps with command syntax
CRPDEPTH	Sets or reads the CRP Depth
CRPDEPTHUSE	Sets or reads indicator to use the CRP Depth

9.12 SVS Commands

Mnemonic	Purpose
LIST	Lists all SVS parameters
HELP	Helps with command syntax
LA	Sets or reads the SVS lever arms (m)
TYPE	Sets or reads type of sound speed sensor

9.13 PRESS Commands

Mnemonic	Purpose
LIST	Lists all PRESS parameters
HELP	Helps with command syntax
LA	Sets or reads the PRESS lever arms (m)
OFFSET	Sets or reads pressure depth offset (m), applied to all pressure input types
KELLRATE	Set the rate a Keller pressure sensor should be sampled

9.14 DVL Commands

Mnemonic	Purpose
LIST	Lists all DVL parameters
HELP	Helps with command syntax
LA	Sets or reads the DVL lever arms (m)
MA	Sets or reads the DVL mounting angles (degrees)
TRIG	Sets the DVL trigger port output
LATENCY	Sets the latency for the DVL
SFERROR	Sets the scale factor error
PREPMAXTLAST	Sets or reads the max time (s) allowed between measurements
PREPMAXACC	Sets or reads the max change in velocity allowed
KFEVEL	Sets or reads the error velocity
MINBEAMS	Sets or reads the settings for number of beams required for measurement to be accepted.
OPFORMAT	Sets or reads the DVL output format
MODE	Sets or reads the mode to be used to decide whether a message should be used for aiding.

9.15 ZUPT Commands

Mnemonic	Purpose
LIST	Lists all ZUPT parameters
HELP	Helps with command syntax
MAXVEL	Sets or reads the maximum velocity

9.16 TSYS Commands

Mnemonic	Purpose
LIST	Lists all TSYS parameters
HELP	Helps with command syntax
SOURCE	Sets or reads the external UTC source to use

Mnemonic	Purpose
ZDA	Sets or reads the port number for GPZDA input
PPS	Sets or reads the port number for 1 PPS input
PPSMODE	Sets or reads the mode of GPZDA and its corresponding 1 PPS
ZDALATENCY	Sets or reads GPZDA latency in seconds
UPDATE	Sets or reads the time in seconds to update the RTC
RST	Resets Time System
DATETIME	Sets or reads the RTC date and time
DATE	Sets or reads the RTC date
TIME	Sets or reads the RTC time
LSZDA	Enables and configures PPS and ZDA generation

9.17 LOG Commands

Mnemonic	Purpose
LIST	Lists all LOG parameters
HELP	Helps with command syntax
MEM	Displays the amount of memory used and free on the SD card
MSG	Lists the messages and their numbers or sets the messages to be logged
NET	Lists the log files going to the ethernet
ROTATE	Sets or reads the time between creating new SD card files

9.18 TRIG Commands

Mnemonic	Purpose
LIST	Lists all TRIG parameters
HELP	Helps with command syntax
ACTIVE	Sets or reads if the trig pulse is active high or low
WIDTH	Sets or reads the trig pulse width
PERIOD	Sets or reads the trig period, if periodic
INPUT	Sets if the trigger is an input or output
START	Sets or reads the start time parameter
GO	Sets or reads if the trigger is running
NRZ	Sets or reads the non-return to zero value

Appendix A – Open Source Software Licence Notices and Terms

The following third party and open source software is used with this product.

A.1 Open Source Software

A.1.1 Tera Term V4.8

Copyright (c) T.Teranishi.

Copyright (c) TeraTerm Project.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
3. The name of the author may not be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES,

INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Definitions

Term	Definition
<cr><lf>	Carriage Return, Linefeed
AHRS	Attitude & Heading Reference System
ASCII	American Standard Code for Information Interchange
AUV	Autonomous Underwater Vehicle
BIST	Built-In Self-Test
C&C	Command & Control
COG	Course Over Ground
CPU	Central Processing Unit
CRP	Central Reference Point
DLE	Data Link Escape
DVL	Doppler Velocity Log
ETX	End of Text
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
HW	Hardware
Hz	Hertz
IMU	Internal Measurement Unit
INS	Inertial Navigation System
kPa	kilopascal
LBL	Long BaseLine
LNAV	Long NAVigation
LSB	Least Significant Bit
LSD	Least Significant Digit
MRU	Motion Reference Unit
MSB	Most Significant Bit
OEM	Original Equipment Manufacturer
PRESS	Pressure Depth
PSI	Pounds per Square Inch
ROV	Remotely Operated Vehicle
RTC	Real Time Calibration
SID	Security IDentifier
SOG	Speed Over Ground
Sonardyne	Sonardyne International Limited and its affiliates.
SPRINT	Subsea Precision Reference Inertial Navigation Technology

Term	Definition
STX	Start of Text
SUSBL	Subsea USBL
TAT	Turn-Around Time
TCVR	Transceiver
TMS	Time System Status
ToV	Time of Validation
USBL	Ultra-Short BaseLine
UTC	Coordinated Universal Time

Global Headquarters, UK
Blackbushe Business Park
Yateley
Hampshire
GU46 6GD
United Kingdom
T. +44 (0) 1252 872288
F. +44 (0) 1252 876100
sales@sonardyne.com

Rio das Ostras, Brasil
Av. Zen lotes 05 e 06
Quadra D
Zen, Rio das Ostras – RJ
CEP 28.890-000, Brasil
T. +55 22 2123 4950
F. +55 22 2123 4951
brasil.sales@sonardyne.com

Singapore, Asia
34 Loyang Crescent
Block B
Singapore
508993
T.+65 6542 1911
F. +65 6542 6937
asia.sales@sonardyne.com

Houston, USA
8280 Willow Place Drive North
Suite 130
Houston,
Texas
77070 USA
T. +1 281 890 2120
F. +1 281 890 7047
usa.sales@sonardyne.com

Aberdeen, UK
Units 12–14,
The Technology Centre
Claymore Drive
Bridge of Don, Aberdeen
AB23 8GD UK
T. +44 (0) 1224 707875
F. +44 (0) 1224 707876
sales@sonardyne.com

24/7 Emergency Helpline
T. +44 (0) 01252 877600

Email Support
support@sonardyne.com

Website
www.sonardyne.com

Twitter
@sonardyne