



USER GUIDE



List of Changes:

Date	Revision	Responsible	Reason for Change
2018-07-10	0.1	T. Poms	- Creation
2018-11-25	0.2	T. Poms	- Revised safety advices - Added section "precompiled binaries" - Revised package installation and package naming convention
2019-05-13	0.3	T. Poms	- Pinout of 40 pin IO header corrected
2019-06-24	0.4	T. Poms	- Added description of antenna connectors - Revised safety advices

Validity:

Unless specified otherwise on www.rooco.eu, this revision of the manual is valid for the following firmware, software and hardware versions, as well as more up-to-date versions.

Firmware
1.2.0

OpenPlotter
1.0.0

Hardware
P770010E01
P770010E02
P770010E03
P770010E04

SUBJECT TO TECHNICAL CHANGES. TYPOGRAPHICAL AND PRINTING ERRORS EXCEPTED.

ALL FIGURES ARE SYMBOLIC PHOTOS.

DISCLOSURE

csoft – Web and IT solutions (hereinafter abbreviated as csoft) has taken great care in compiling the information presented in this document and is updating the contents continually. Despite this, csoft assumes no responsibility whatsoever for any information which is depicted incorrectly or incomplete. csoft excludes liability for any direct or indirect mistakes and damage/loss resulting from the use of this manual.

This document can be changed at any time and without prior notice by csoft.

Visit our website (<https://www.rooco.eu>) to download the most up-to-date version of this user manual.

COPYRIGHTS

The entire content of this document is copyright protected. All rights are reserved. Any utilization requires express permission from csoft and/or the respective bearers of the copyrights. Noncompliance shall entail an obligation to compensate for damages.

WARRANTY CONDITIONS

csoft rectify without charge defects to your Moitessier HAT which are attributable to material and/or manufacturing deficiencies, and which are communicated to csoft within the warranty period (two years from the date of purchase).

It shall be at the discretion of csoft to decide upon the actions taken to rectify a fault. Parts are repaired, or the product is swapped out, on the basis of a replacement against an equivalent product which is not necessarily the same model, whereby it is a new product or a reconditioned product corresponding to a new product as regards functionality. The warranty for repaired or replaced parts is assumed for the remaining period of the warranty term. All original parts replaced are transferred to the ownership of csoft. New parts and replacement parts are transferred to the ownership of the customer.

The warranty program does not apply for csoft products whose serial numbers have been removed, defaced or changed, or for when the customer has opened up the product without authorization (if applicable). The warranty also does not cover the following damage:

- Damage caused by accident, misuse or improper use, especially in the event of noncompliance with the operating instructions for the product
- Damage from the use of parts not manufactured or sold by csoft
- Damage from modifications performed which were not approved in writing beforehand by csoft
- Damage from services not rendered by csoft or authorized representatives of csoft
- Damage caused by transportation, carelessness, fluctuations or outage of the power supply, force majeure, lightning strike, liquids, fire or operating environment
- Damage from normal wear and tear

- Damage/loss resulting from a reconfiguration of the product delivered (applicable for hardware and software)
- Damage/loss from the specification/reconfiguration of passwords
- Misuse, and use of the device other than as intended, as well as incorrect installation
- Damage/loss from use of the product which is not within the specification

Additional fees are charged for work, transportation and parts for services rendered by csoft in conjunction with the rectification of such deficiencies or damage attributable to one of the aforementioned reasons for exclusion.

Requirements for asserting claim to this warranty program

The following requirements must be satisfied by the customer for asserting claim to services laid down in this warranty program:

- To assert claim to the warranty, the customer must contact csoft within the warranty period
- The customer must verify the purchase date (indicating the start of the warranty period) by submitting the original purchase receipt or a copy thereof
- The customer must make available an unambiguous fault description and carry out fault analyses in line with the instructions
- The customer must send in the complete product as delivered, including any hardware and other media
- The customer must ensure that the product is packaged appropriately for transportation

Table of Contents

1	INTRODUCTION	7
2	ABOUT THESE INSTRUCTIONS	8
3	SAFETY ADVICES	9
4	CONTENT OF DELIVERY	11
5	HARDWARE OVERVIEW	12
5.1	Pinout – 40 Pin Raspberry Pi IO Header.....	12
5.2	Pinout – HAT headers.....	13
5.3	Powering the System.....	14
6	ANTENNA SYSTEMS SUPPORTED	15
7	INSTALLATION	17
7.1	Installing the HAT on the Raspberry Pi.....	17
7.2	Connecting the Antennas.....	17
7.3	Removing the HAT from the Raspberry Pi.....	18
8	USING THE HAT WITH OPENPLOTTER	19
8.1	Installation of OpenPlotter	19
8.2	Installation of the Moitessier HAT Software	20
8.3	Precompiled Binaries	21
8.4	Configuring AIS/GNSS Reception	22
8.5	Configuring Compass, Heel and Trim Reception	22
8.6	Configuring Pressure and Temperature Reception.....	23
8.7	Displaying Data in OpenCPN	26
8.8	Displaying Data in a Wi-Fi enabled App.....	26
8.9	Configuring the Moitessier HAT	27
9	SOFTWARE COMPILATION AND INSTALLATION	28
9.1	Getting the CompilerTo	28
9.2	Getting the Sources.....	28
9.3	Updating the Sources.....	29
9.4	Compilation.....	29

9.5	Package Installation	30
9.6	Virtual Machine	32
10	COMMUNICATING WITH THE HAT	34
10.1	Reading AIS and GNSS Data.....	34
10.2	Reading Sensor Data.....	35
10.3	Controlling/configuring the HAT.....	35
11	STATUS LEADS.....	37
12	RECEIVING NMEA DATA OVER WI-FI.....	39
12.1	iSailor	39
12.2	NV Chart	40
12.3	iNavX	41
13	CONTACT AND SUPPORT INFORMATION	42
14	PART NUMBERING SCHEME.....	43
15	TECHNICAL SPECIFICATION.....	44

1 Introduction

The Moitesser HAT is named after the famous sailor Bernard Moitessier. It provides full, open-source marine navigation features for the Raspberry Pi (and compatible) and is officially supported by OpenPlotter.

Key facts at a glance

- Fully compatible with Raspberry Pi models supporting 40-pin IO header
- High-sensitivity dual channel AIS receiver with SMA antenna connector (better than -112 dBm)
- High-performance GNSS receiver with integrated patch antenna. An external antenna is supported via BNC connector.
- 3 status LEDs (AIS status, GNSS status, error)
- Barometric pressure, compass, heel and trim. Optional humidity and temperature (only reasonable for standalone usage). Sensors are directly accessible via Raspberry Pi.
- IO headers (optional) to interface with spare GPIOs of the Raspberry Pi and the HAT's microcontroller (e.g. software emulated I²C)
- UART signals of Raspberry Pi available on header (optional)
- Data communication via SPI (AIS, GNSS and meta data) and via I²C (sensor data). Data accessible via device driver and device file.
- Supports ID EEPROM and automatic device tree loading
- Firmware upgradeable via Raspberry Pi
- OpenPlotter compatible (<http://sailoog.com/openplotter>)

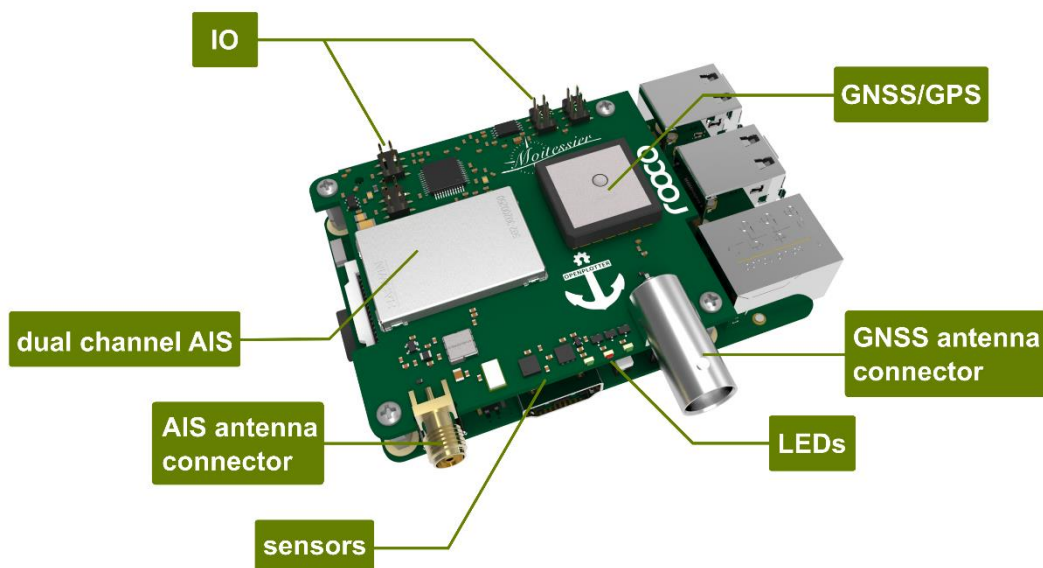


Figure 1: Moitessier HAT attached on a Raspberry Pi 3 Model B

2 About these Instructions

Read the safety instructions carefully before using the device. Heed the warnings in the operating instructions.

Always keep the operating instructions to hand. If you sell or pass on the device, ensure the user guide stay with the device.

Even if the HAT itself might be used for any Raspberry Pi compatible hardware, this user guide focuses only on the use with the Raspberry Pi (particularly model 3B).

Symbols used in this document:



This symbol makes reference to potential risks of injury or hazards to health.



This symbol makes reference to important information.



This symbol makes reference to actions that may entail damage to the product itself or other material assets.

3 Safety Advices




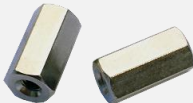


- (1) Damage to the device can occur as the result of improper handling or electrostatic discharge (ESD). Always handle the device with care to avoid damage to electrostatically sensitive components.
Only handle by the edges of the device to minimize the risk of electrostatic discharge damage.
- (2) Read the technical specification to operate the device in a safe setup.
- (3) Avoid handling the device while it is powered.
- (4) Do not connect the device to the Raspberry Pi while it is powered.
- (5) Use the device with compatible hardware only.
- (6) Do not alter the construction or design.
- (7) Do not expose the device to water, moisture or place it on a conductive surface whilst in operation.
- (8) Do not expose the device to heat from any source. It is designed for reliable operation at normal ambient room temperature. Do not expose the device to direct sunlight.
- (9) Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.
- (10) Do not power the device directly through pin 1 of the 40 pin IO header, if not used in standalone mode. Neither power the Raspberry Pi through pins 2 and 4.
- (11) The device must be powered by a current limited power supply. See the technical specification for proper limit values.
Use CE certified power supplies only.
- (12) Do not repair the device on your own.
- (13) Do not exceed the input voltage specification of the digital input pins (GPIO).
- (14) Do not short the output pins of the device. Never reconfigure or drive pins manually used by the HAT. Use the ID EEPROM of the HAT and the provided device driver to initialize the HAT by the Raspberry Pi. See the listing of the reserved GPIO pins in the technical specification.
- (15) Do not drive the antenna inputs with an external DC voltage.
- (16) You must not share the VHF antenna with other radio equipment, without using a splitter, that physically decouples the receiver from any transmitter while transmission is in progress. High input power will damage the device.

- (17) The device has no lightning protection. A high enough energy event at the antenna inputs can damage beyond repair all connected electronic parts. Injuries to humans can happen.
- (18) Do not use adapters directly on the antenna input of the HAT to interface with different connector types. If required, use pigtail adapters to reduce mechanical force on the antenna connectors.
- (19) The distance between the antennas connected to the device and other radio equipment (transmitters) must be at least 4 meters. However, this is just a guide value, the required distance depends on the output power of the transmitter. High radio transmitters power might damage the device. The RF input power at the antenna connector must not exceed 0dBm (1mW).
- (20) Do not solely rely on this device to avoid collisions. Also use your navigation experience and your five senses.

This device on no account takes the place of good seamanship.

4 Content of Delivery

Check the contents of the delivery and check your purchased items for possible transport damage before using the Moitessier HAT.

<p>Moitessier HAT</p>	
<p>2 pieces of spacers (10mm, M2.5)</p>	
<p>4 pieces of mounting screws (M2.5)</p>	
<p>Quick start guide</p>	
<p>Safety information</p>	<div data-bbox="995 1529 1311 1841"> <p>Safety Information</p> <p>⚠ WARNING: Read all safety advices and instructions. Non-observance of the safety advices and instructions could result in injury to people and/or damage to property.</p> <p>Intended use Use at your own risk. The Moitessier HAT (hereinafter referred as "device") is intended to provide informational data. Do not rely upon this information for any purpose without independent equipment.</p> <p>General safety advices Do not expose the device to water, moisture or place it on a conductive surface whilst in operation. Do not expose the device to heat from any source. It is designed for reliable operation at normal ambient room temperature. Take care whilst handling to avoid mechanical or electrical damage to the printed circuit board and connectors.</p> </div>

5 Hardware Overview

5.1 Pinout – 40 Pin Raspberry Pi IO Header

3.3V	1 ● 2 ●	5V	
I ² C SDA (GPIO 02)	3 ● 4 ●	5V	
I ² C SCL (GPIO 03)	5 ● 6 ●	GND	
GPIO 04	7 ○ 8 ●	TXD (GPIO 14)	
GND	9 ● 10 ●	RXD (GPIO 15)	
OUTPUT (GPIO 17)	11 ● 12 ●	OUTPUT (GPIO 18)	
INPUT (GPIO 27)	13 ● 14 ●	GND	
OUTPUT (GPIO 22)	15 ● 16 ●	INPUT (GPIO 23)	
3.3V	17 ● 18 ●	OUTPUT (GPIO 24)	
SPI MOSI (GPIO 10)	19 ● 20 ●	GND	
SPI MISO (GPIO 09)	21 ● 22 ○	GPIO 25	
SPI CLK (GPIO 11)	23 ● 24 ●	SPI CS (GPIO 08)	
GND	25 ● 26 ○	GPIO 07	
I ² C SDA ID EEPROM	27 ● 28 ●	I ² C SCL ID EEPROM	
GPIO 05	29 ○ 30 ●	GND	
GPIO 06	31 ○ 32 ○	GPIO 12	
GPIO 13	33 ○ 34 ●	GND	
GPIO 19	35 ● 36 ●	GPIO 16	
GPIO 26	37 ● 38 ●	GPIO 20	
GND	39 ● 40 ●	GPIO 21	

Legend

- power pin
- ground pin
- pin used exclusively by HAT
- shared pin
- pin not used by HAT but accessible via optional header on the HAT
- unused pin

INPUT/OUTPUT direction seen from Raspberry Pi

Figure 2: pinout of the Raspberry Pi 3 GPIO header



USE PINS EXCLUSIVELY ASSIGNED TO THE HAT ONLY FOR THEIR INTENDED PURPOSE. YOU WILL DESTROY THE HAT OR THE RASPBERRY PI, IF PINS ARE CONFIGURED INCORRECTLY.

I²C and SPI bus can be shared with other hardware. Keep in mind that that this is not applicable for the chip select used with the SPI bus, which is exclusively used by the HAT.

Pins marked blue are not used by the HAT itself, but are accessible for extension purpose on optional headers on the HAT (see figure 3).

5.2 Pinout – HAT headers

The Moitessier HAT features up to three optional headers, depending on the part number. Signals labeled with the prefix *DEBUG* are routed to the microcontroller of the HAT and are for debugging purpose only, mainly used for testing during production.

On header J503 and J701 the blue marked signals according to figure 2 are accessible. These signals are physically routed directly to the Raspberry Pi.

Header J702 is used for debugging and testing only and should not be used by the end customer.

The UART of the Raspberry Pi (GPIO15, GPIO14) can be accessed on J701. Be aware that this UART might be shared with the onboard Bluetooth module on the Raspberry Pi. You might need to change your boot configuration to use this interface.



DO NOT DRAW MORE THAN 15MA FROM THE 3.3V PIN. THE VOLTAGE REGULATOR OF THE RASPBERRY PI MIGHT REACH ITS CURRENT LIMITATION. AN EXCESSIVE VOLTAGE DROP MIGHT CAUSE A MALFUNCTION OF THE SYSTEM.

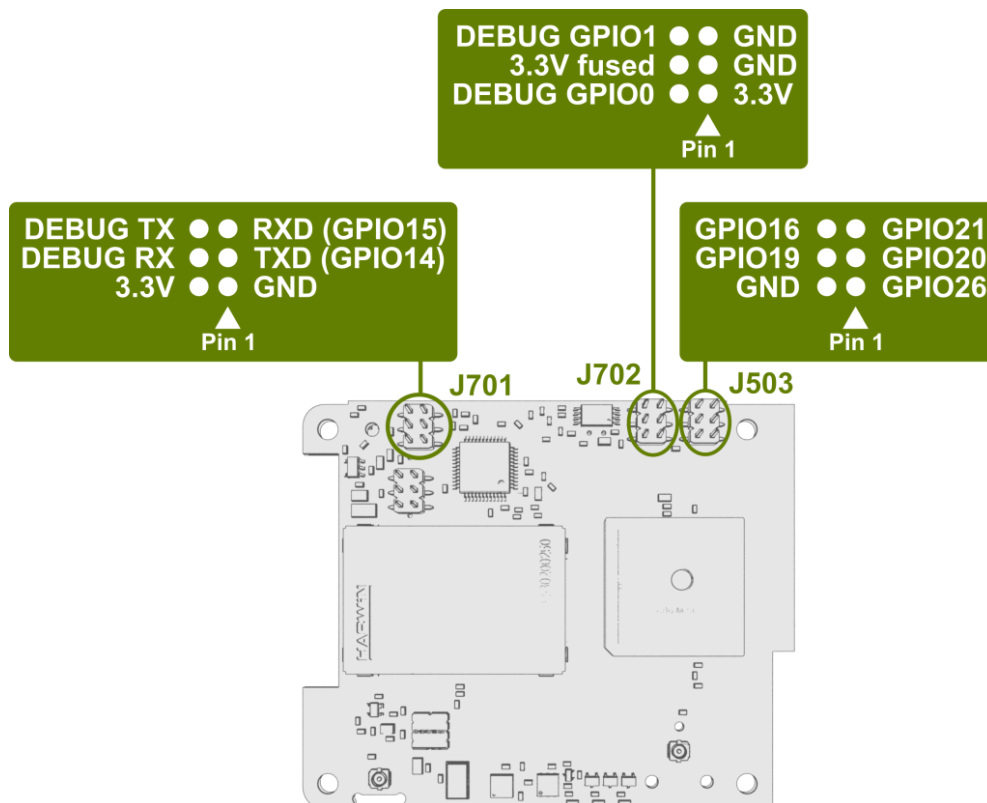


Figure 3: pinout of the HAT headers

5.3 Powering the System

The Moitessier HAT is powered directly via the 3.3V pins of the 40-pin IO header. To supply the Raspberry Pi use only CE certified mains power plugs or 5V converters that can deliver at least 2A.

Select the proper power source with care. Use only low-noise and high-quality supplies as this have a direct impact on the reception performance of the Moitessier HAT.

6 Antenna Systems supported

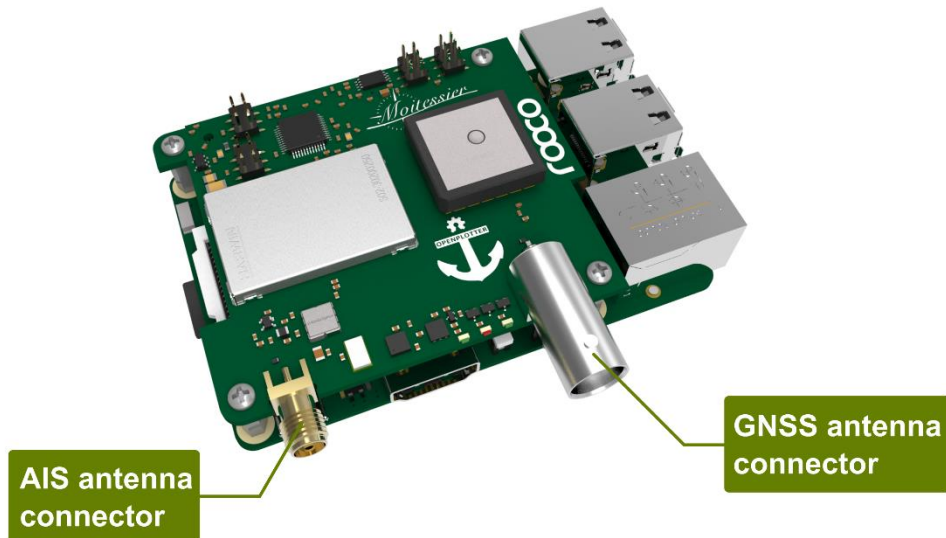


Figure 4: Antenna connectors of the HAT

AIS Antenna

The Moitessier HAT supports all popular VHF/AIS antennas. Please note the following features when selecting an antenna:

- 50 Ohm impedance
- SMA male connector for direct connection, or any other connector using a proper pigtail adapter
- Frequency range at least 161.95 MHz to 162.05 MHz
- RG 174 coaxial cable or better



The coaxial cable attached to the SMA connector should have a maximum outside diameter of 3.7 mm. Larger diameters might cause mechanical force to the antenna connector. It is recommended to use pigtail adapters.

A suitable splitter also enables the Moitessier HAT to share the VHF antenna of other radio equipment on a ship. Use splitters only that physically decouple the Moitessier HAT from any transmitter while transmission is in progress.

GNSS Antenna

Your device has an internal patch antenna. If it is not possible to fit the HAT with an unobstructed view of the sky (such as below deck), an external GNSS antenna is required.

Use a standard, active GNSS antenna that is fitted with a BNC connector.

Connectors



Figure 5: Supported antenna connectors



For the best receiving performance, ensure that the cable lengths of the antennas are as short as possible.



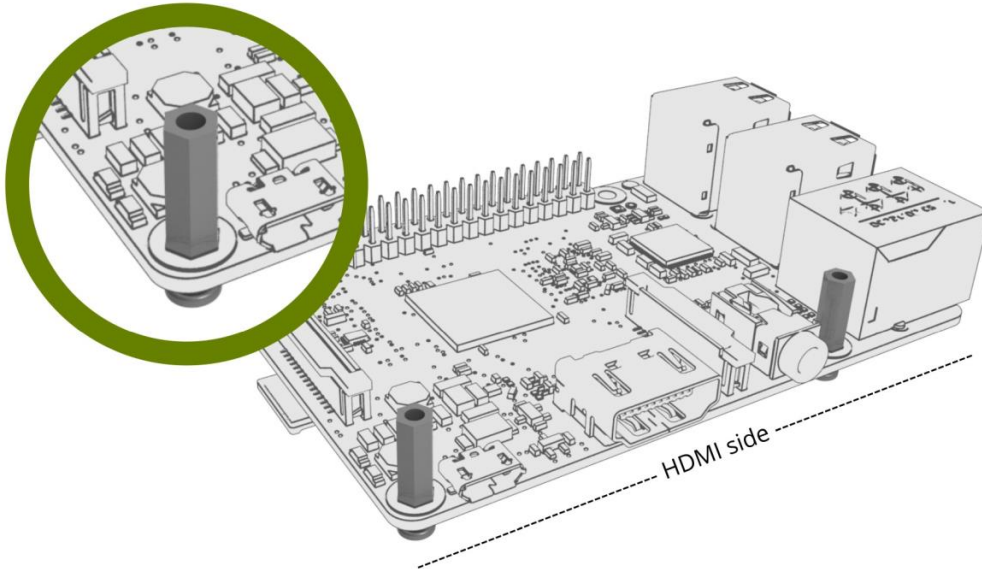
Do not use adapters directly on the antenna input of the HAT to interface with different connector types. If required, use pigtail adapters to reduce mechanical force on the antenna connectors.



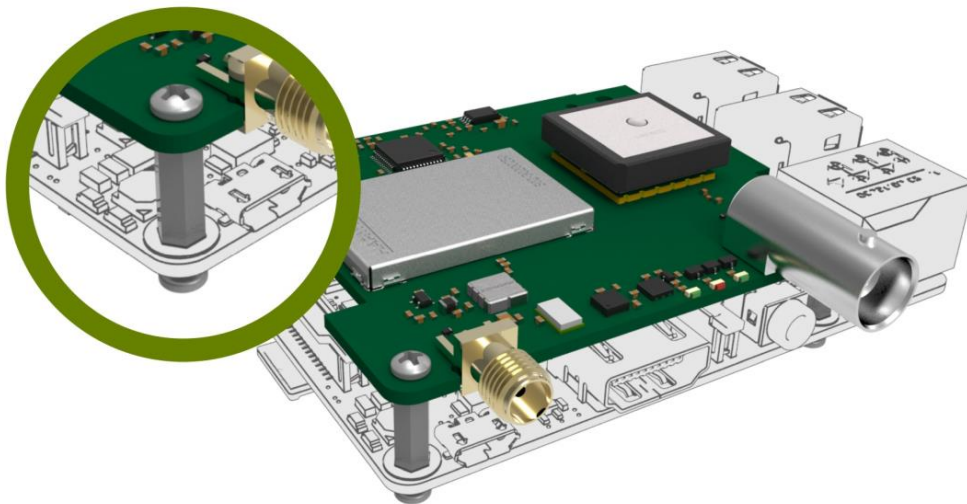
7 Installation

7.1 Installing the HAT on the Raspberry Pi

(1) Fix the two spacers with the screws on the Raspberry Pi (HDMI side only).



(2) Attach the HAT and screw Raspberry Pi and HAT together.



7.2 Connecting the Antennas

GNSS

Mount the external GNSS antenna in a raised position with an unobstructed view of the sky. 360° horizontal visibility and a vertical view angle greater than 45° should be observed for best possible performance. Other electronic devices can interfere with the GNSS signal. Ensure that the antenna is as far away as possible from radar and/or the VHF antenna of the radio equipment. A distance of at least four meters must be maintained.

AIS

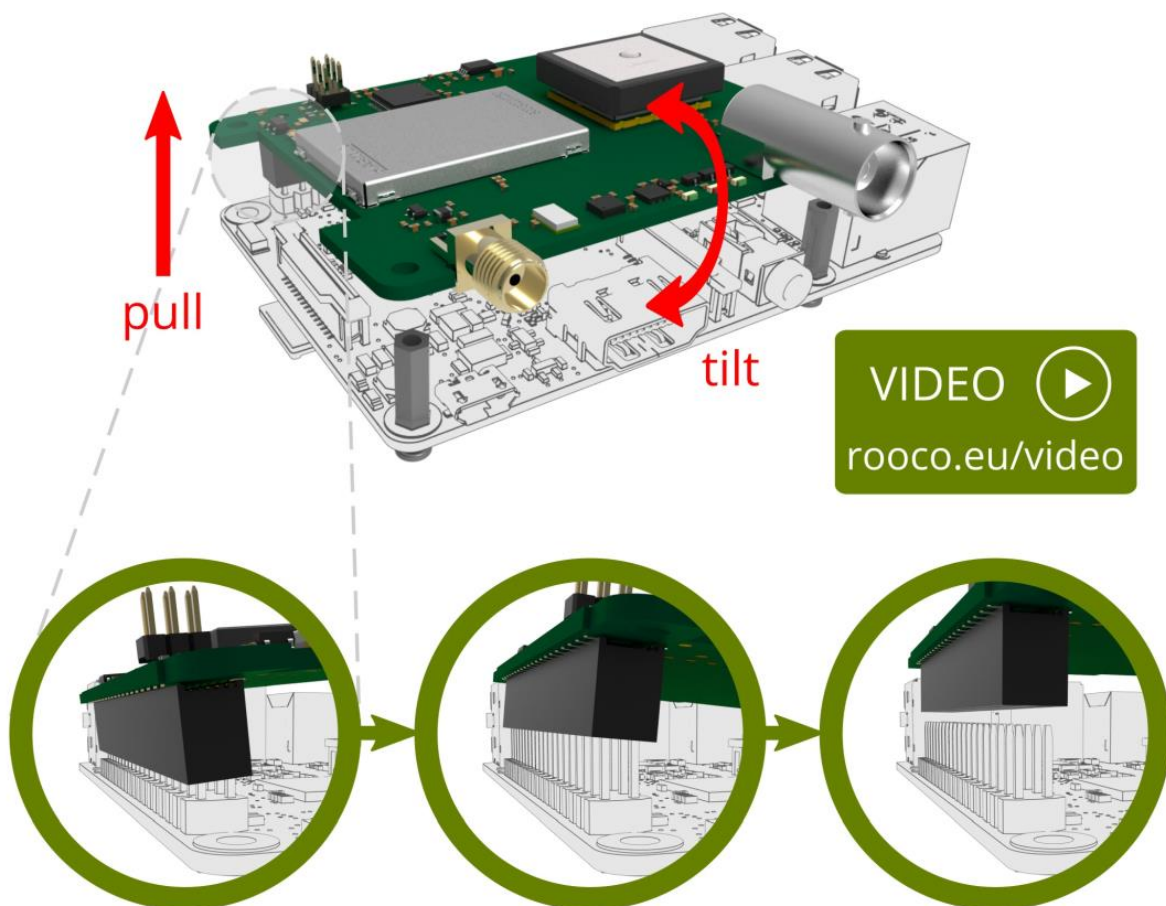
Mount the AIS antenna as high as possible to attain the best reception results. Keep a minimum distance of four meters to other VHF antennas. A splitter must be used if you want to share the antenna with a VHF radio equipment. Use active splitters only, that physically decouples the receiver from any transmitter while transmission is in progress.



THE MOITESSIER HAT MUST NOT BE CONNECTED DIRECTLY TO THE ANTENNA OUTPUT OF A VHF RADIO EQUIPMENT OR ANY SIMILAR TRANSMITTING DEVICE. THIS WOULD DAMAGE YOUR HAT AND/OR RASPBERRY PI BEYOND REPAIR.

7.3 Removing the HAT from the Raspberry Pi

- (1) Remove the screws.
- (2) Tilt and pull the HAT gently until you can remove it completely.



AVOID BENDING THE PINS OF THE 40-PIN IO HEADER. ROTATE ABOUT THE SHORTER SIDE OF THE RASPBERRY PI.

8 Using the HAT with OpenPlotter

“OpenPlotter is a combination of software and hardware to be used as navigational aid on small and medium boats. It is also a complete home automation system onboard. It works on ARM computers like the Raspberry Pi and is open-source, low-cost and low-consumption. Its design is modular, so you just have to implement what your boat needs.”¹

This chapter will focus solely on how to use the Moitessier HAT with OpenPlotter. It does not describe the functionality of OpenPlotter in detail, especially not those functions that are not directly related to the HAT. Additional information about OpenPlotter can be found at <http://www.sailoog.com/openplotter> and <https://docs.sailoog.com/openplotter-v1-x-x/>



Even though the Moitessier HAT can be used without OpenPlotter, it is not recommended to use other third-parts tools. You might struggle to read AIS/GNSS and sensor data with your navigation software, if you need to configure all the tools on your own.

However, chapter 9 might give you some advice to compile the sources and to create your own Raspbian package. Chapter 10 gives you a short introduction how to communicate with the HAT without OpenPlotter.

8.1 Installation of OpenPlotter

- (1) Download the latest version at https://archive.org/details/openplotter_v1.0.0_noobs.
- (2) Unzip the file and copy the extracted content to a FAT32 formatted SD card with at least 8 GByte memory size.
- (3) Insert the SD card into your Raspberry Pi and power on the device.
- (4) The OpenPlotter installer starts the installation. There is no need for any user interaction. This will take several minutes.

It is recommended to use a display or monitor connected to the HDMI port of the Raspberry Pi, so you can follow the installation progress. Afterwards you might want to use a VNC viewer² on a tablet, PC or laptop to access the Raspberry Pi.

Login credentials:

- User: pi
- Password: raspberry

Default Wi-Fi credentials:

- SSID: openplotter
- Wi-Fi password: 12345678
- IP address: 10.10.10.1

¹ <http://sailoog.com/openplotter>

² <https://www.realvnc.com>



It is recommended, that you check for OpenPlotter updates regularly to keep your system up to date. Whenever you update OpenPlotter, it might be necessary to update the Moitessier HAT software as well.

8.2 Installation of the Moitessier HAT Software

Start OpenPlotter using the icon in the menu bar. Open *Moitessier HAT* in the tools selection and select *Start* in the popup window.

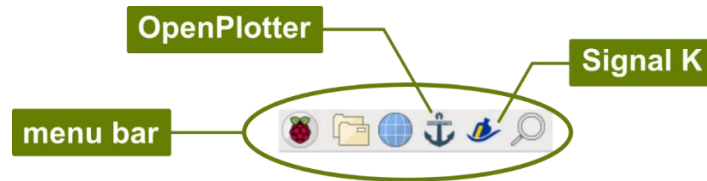


Figure 6: Raspbian menu bar

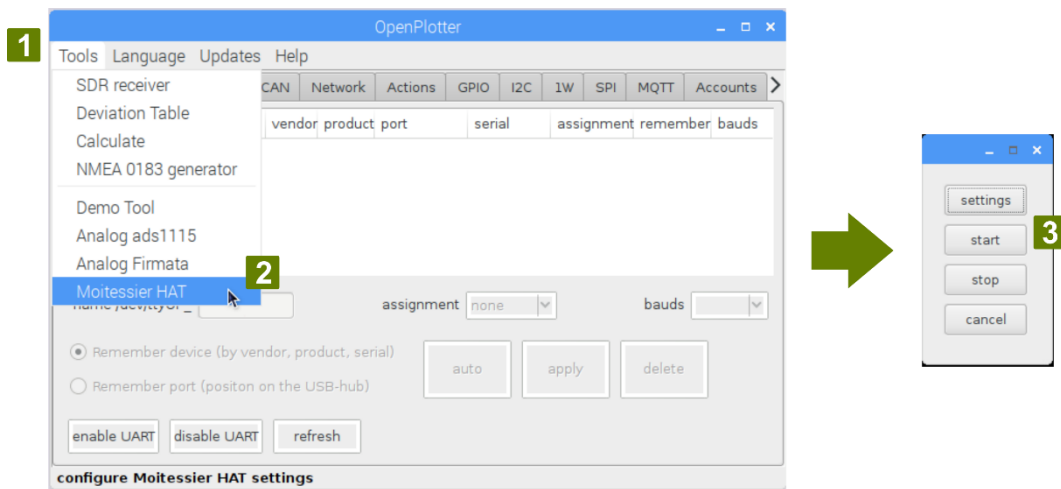


Figure 7: OpenPlotter - tool selection menu

Select the tab *Install* and check your kernel version. Only the first 3 figures are important, in this case *4.14.71*. Select one of the available packages that match the kernel version.

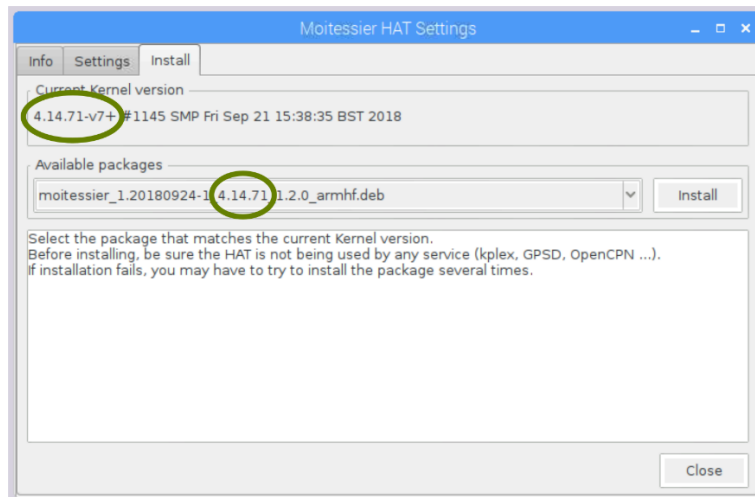


Figure 8: OpenPlotter - Moitessier HAT package installation

Click the *Install* button to start the installation. A terminal window will open and gives you information on the installation progress. The system will reboot afterwards automatically.



Ensure that you install only packages that match the kernel version. The software will not work properly, if using different versions.

If a wrong version of the Moitessier HAT software is installed, the serial device created for the HAT will be highlighted in red. Keep in mind, that you need to install the proper package related to the running kernel on your system. Whenever you update OpenPlotter, it might be necessary to update the Moitessier HAT software as well.

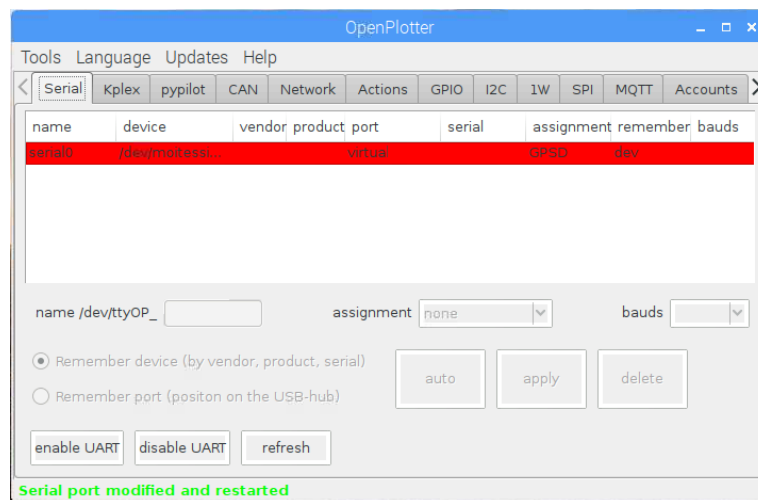


Figure 9: OpenPlotter – wrong Moitessier HAT version installed

8.3 Precompiled Binaries

OpenPlotter is distributed with precompiled binaries. However, the Moitessier HAT software is continuously improved and the include binaries in OpenPlotter might get outdated over time. To keep your system up to date, you might want to compile the sources at your own (see chapter 9), or you can download the latest binaries from <https://get.rooco.tech/moitessier>.

Section 9.5 will explain how to install packages that are not included in the OpenPlotter distribution.

8.4 Configuring AIS/GNSS Reception

After reboot, start OpenPlotter again. A new device appears in the *Serial* tab. Select the new device and press the *auto* button. The system will find the best settings for you and will fill the proper fields of the tab. Use the *apply* button to save the settings.

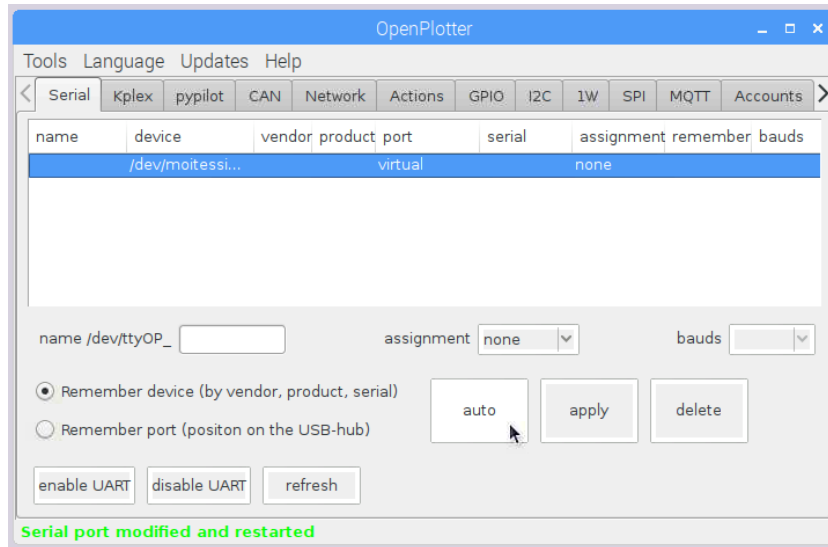


Figure 10: OpenPlotter – serial configuration tab

8.5 Configuring Compass, Heel and Trim Reception

Select the tab *pyplot* and use either *basic autopilot* mode or *IMU only*, whether you have a *pyplot* installation or not.

Check the type of data you want to enable – heading or/and trim and heel (pitch, roll). A translation rate of 0.5 or 1 second should be enough.

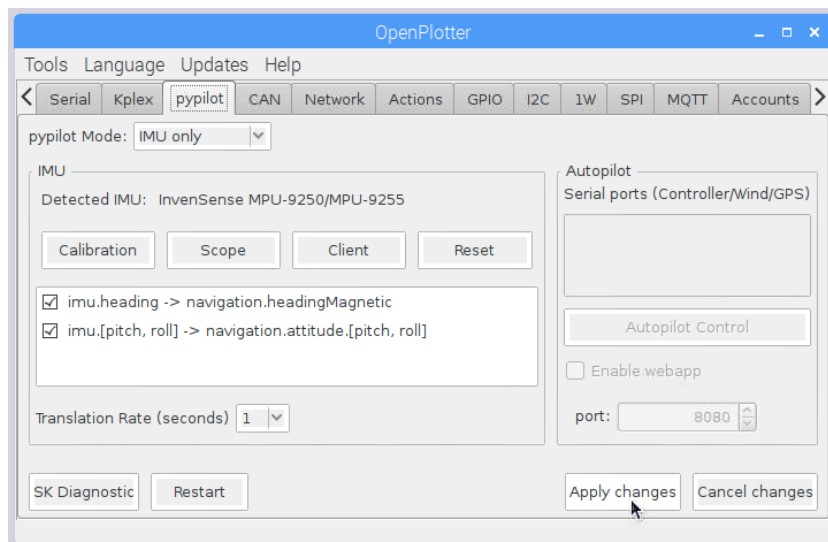


Figure 11: OpenPlotter – pyplot/IMU configuration tab

8.6 Configuring Pressure and Temperature Reception

Select the tab *I2C*, press the *add* button, select the MS5607-02BA03 sensor and apply the settings by clicking the *OK* button. One pressure and one temperature device will be listed afterwards.

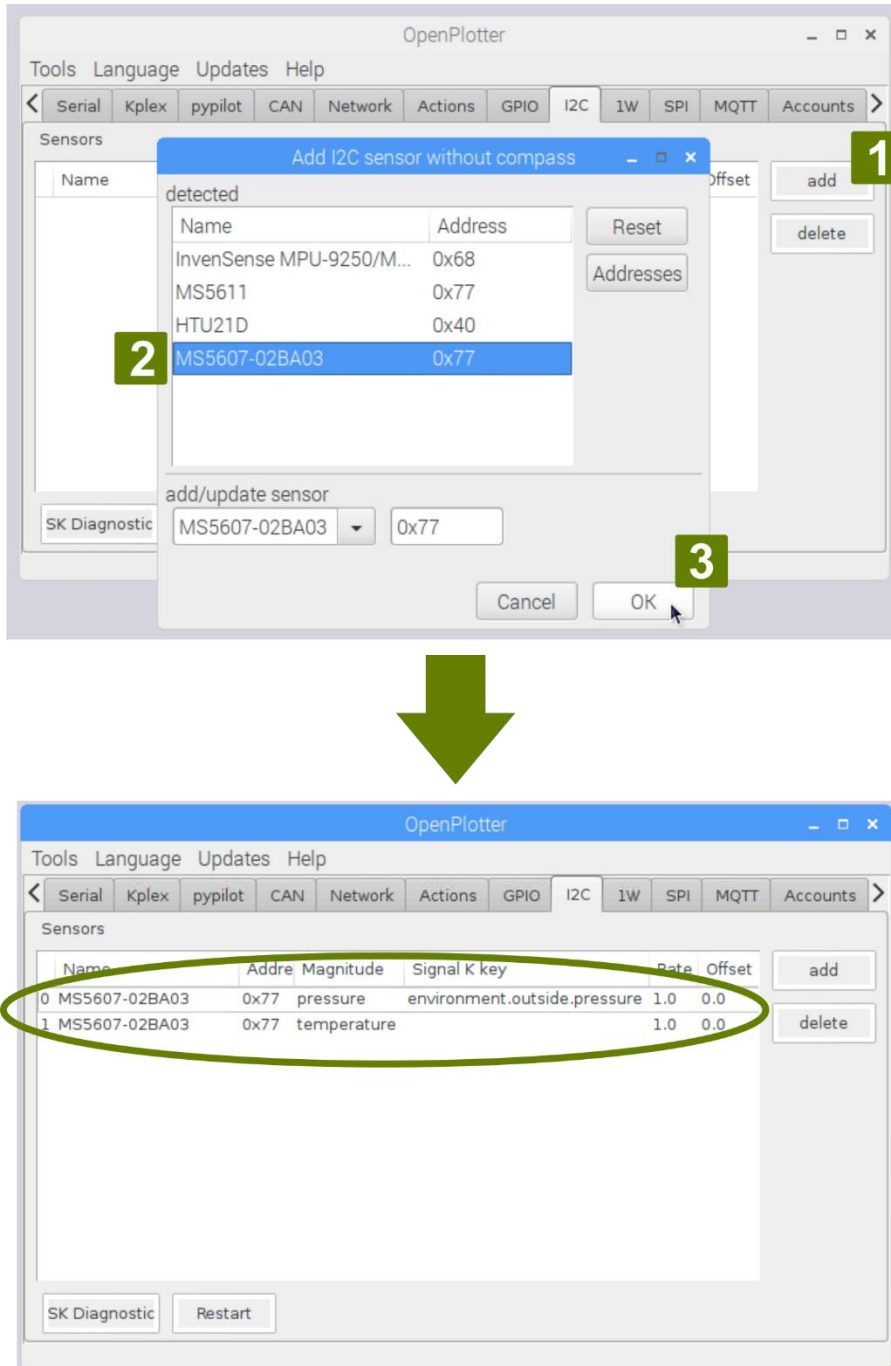


Figure 12: OpenPlotter – adding I²C sensor

Some features will have a Signal K key pre-assigned when its function is obvious (e.g. pressure). Some features with more than one possible Signal K key assignment will be blank.

Double click on the second line in the list (*temperature*) and edit the Signal K key by selecting group *environment* and the key *environment.inside.temperature*. In case of the

temperature sensor you probably do not need to send data every second. You can define the update interval in the *Rate* field. If you think that your sensors could have any deviation, use *Offset* to correct it. Apply the setting by clicking the *OK* button.

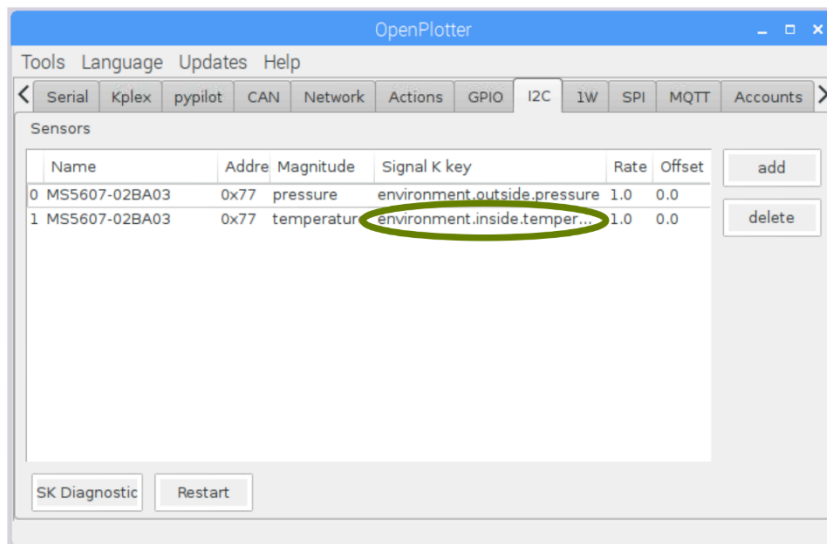
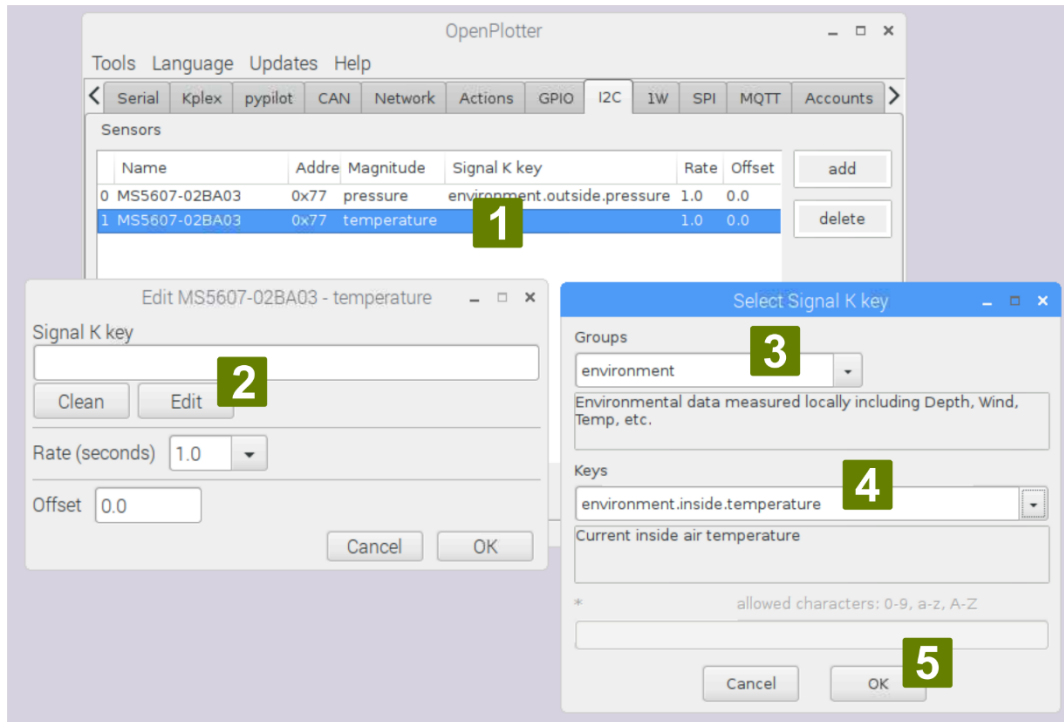


Figure 13: OpenPlotter – assigning Signal K key to I²C sensor



Temperature readings cannot be considered reliable, because it is affected by the temperature of the Raspberry Pi. However, it is a good way to get the temperature of your system.

Open the Signal K web application using the Signal K icon in the Raspbian menu bar (see figure 6). Log into the server application using the *Login* button in the upper right corner of the web application.

Login credentials:

- User: openplotter
- Password: openplotter

Go to *Server* → *Plugin Config* → *Convert Signal K to NMEA0183* and check *Active*. Scroll down and check *XDR (Barometer)* and *XDR (Air temperature)* as well.

To apply the settings, click the *Submit* button.

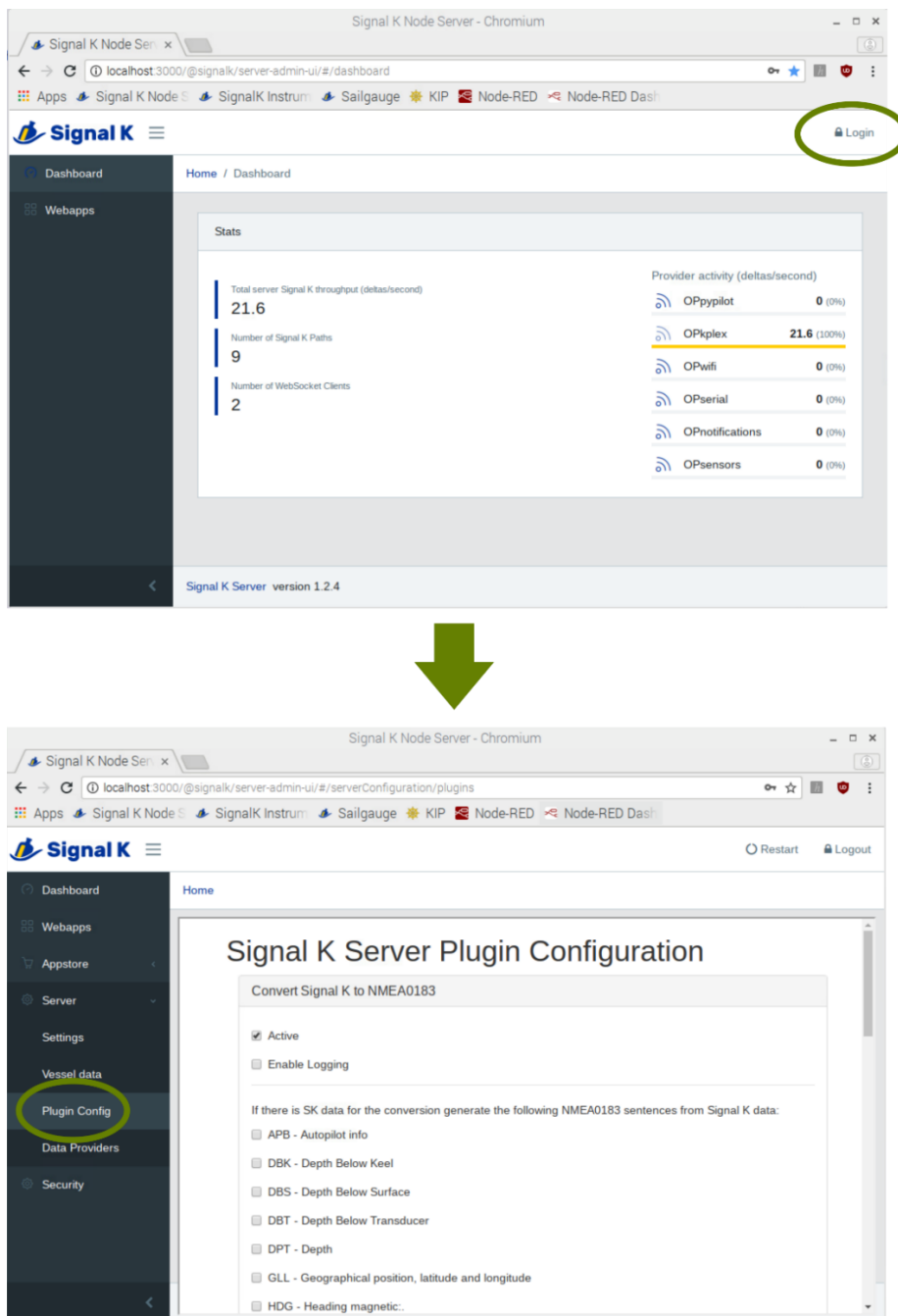


Figure 14: Signal K – plugin configuration

8.7 Displaying Data in OpenCPN

Start OpenCPN and open the settings window. Go to the tab *Plugins* and enable the *Dashboard*.

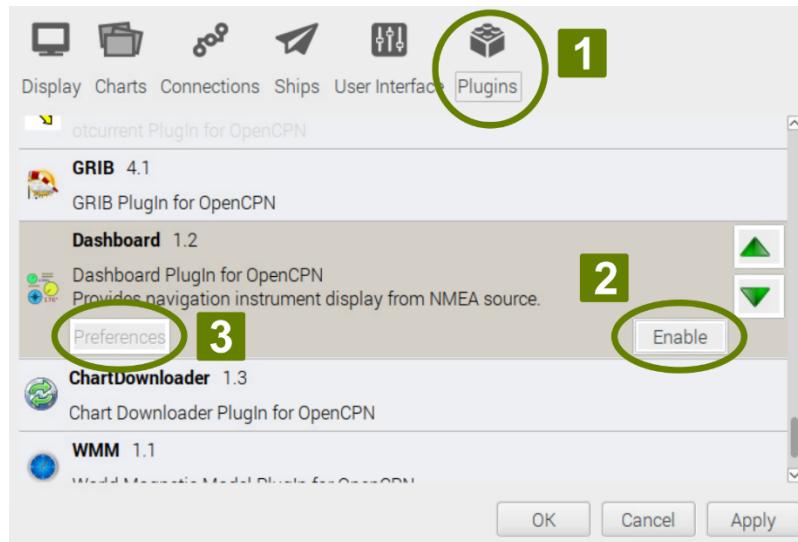


Figure 15: OpenCPN – plugin/dashboard configuration

Configure the preferences/appearance of the dashboard by enabling the required instruments. You might want to use the following:

- GPS Status
- Air Temp.
- Mag HDG
- Pitch
- Heel
- Barometric pressure
- Barometric history

OpenPlotter should now display AIS targets, GNSS status and the proper sensor information (magnetic heading, pressure, temperature, etc.).

8.8 Displaying Data in a Wi-Fi enabled App

If you want to use OpenPlotter as a headless system without display, you can receive the NMEA data on any Wi-Fi enabled device (e.g. tablet, smartphone, PC, laptop, etc.). The device can use apps like NV Charts, iSailor, iNavX, OpenCPN, etc.

See chapter 11 for more information on this topic.

8.9 Configuring the Moitessier HAT

Open *Moitessier HAT* in the tools selection of OpenPlotter and select *Settings* or *Start* in the popup window.

Select the tab *Settings*, change the configuration as required and click the *Apply changes* button.

The *frequency*, *afcRange* and *tcxoFreq* settings are for experts only. They might be adjusted for testing purpose only.

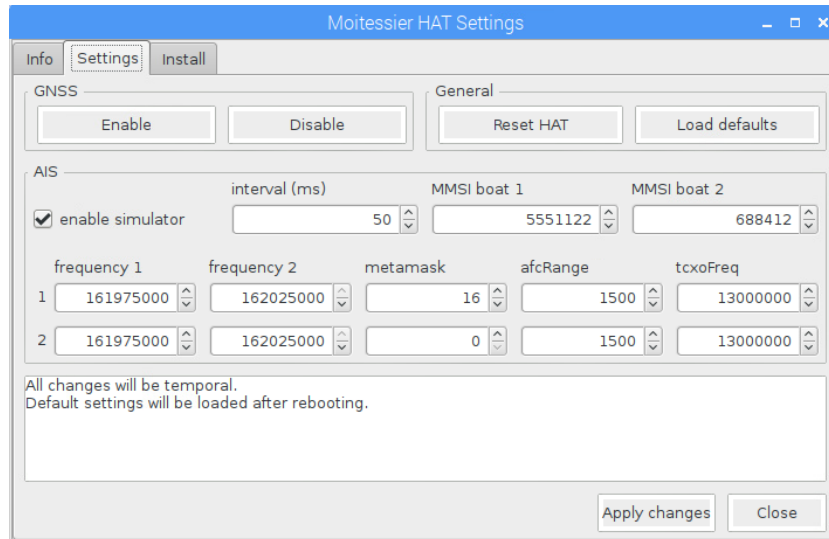


Figure 16: OpenPlotter – Moitessier HAT settings



The configuration is not permanently stored in the HAT, so you will need to reconfigure the HAT whenever you reset the HAT or the Raspberry Pi.

Activate the AIS simulator to generate simulated AIS data. You can use this functionality to check the communication between the Moitessier HAT, the Raspberry Pi and your navigation software without having to receive real AIS data. The simulator generates two AIS targets in the Adriatic Sea/Venice.

The configuration can also be modified via a terminal. See section 10.3 for further information.

9 Software Compilation and Installation



This chapter describes how to compile the sources and how to create an installable Raspbian package. It is intended for advanced users, that want to add functionality and/or modify the source code provided by Rooco.

You can skip this section, if you use OpenPlotter and the included precompiled binaries for the Moitessier HAT.

To compile the sources for the Moitessier HAT you need at least the kernel sources for the Raspberry Pi and a proper toolchain. The following sections will describe how to setup the development machine (hereinafter referred to as *host*). You can skip these sections and continue with 0, if you want to use our prepared virtual machine.

9.1 Getting the Compiler

Open a terminal shell on your machine and execute the following commands, to install the official Raspberry Pi cross toolchain.

```
host> mkdir -p ~/Desktop/Raspberry
host> cd ~/Desktop/Raspberry
host> sudo apt-get install git bc
host> git clone https://github.com/raspberrypi/tools toolchain
host> echo PATH=$PATH:~/Desktop/Raspberry/toolchain/arm-bcm2708/gcc-
linaro-arm-linux-gnueabi-hf-raspbian/bin >> ~/.bashrc
host> source ~/.bashrc
```

Check if the system recognizes the compiler.

```
host> which arm-linux-gnueabi-hf-gcc
```

The command output should show something like this (depending on your installation path):

```
/home/development/Desktop/Raspberry/toolchain/arm-bcm2708/gcc-linaro-arm-
linux-gnueabi-hf-raspbian/bin/arm-linux-gnueabi-hf-gcc
```

9.2 Getting the Sources

Kernel

The Moitessier HAT uses a dynamically loadable kernel device driver module. To cross compile this module, the kernel sources needs to be referenced. Keep in mind, that the kernel on your target, must have the same version as the kernel sources used for cross compilation. If the target kernel is based on a different kernel source tree, the module loading will fail. The same applies for the toolchain used for kernel creation. It must be the same to compile the device driver. You'll get an *invalid module format* error when loading the driver, if you do not take that into account.

```
host> cd ~/Desktop/Raspberry
host> git clone https://github.com/raspberrypi/linux kernel
```

Moitessier HAT

The source repository is available at GitHub.

```
host> cd ~/Desktop/Raspberry
host> git clone https://github.com/mr-rooney/moitessier.git
host> cd moitessier
```

Only once after cloning you need to initialize the cloned submodules.

```
host> git submodule update --init --recursive
```

To update the sources of the submodules at a later time do the following.

```
host> git submodule update
```

The following steps need to be repeated for each submodule. `<SUBMODULE>` is a placeholder for the proper submodule name. To list the submodules use the shell command `ls`.

```
host> cd <SUBMODULE>
host> git checkout master
host> cd ..
```

9.3 Updating the Sources

The kernel as well as the Moitessier HAT sources are actively maintained, so code will change in the course of time.

Before you compile the sources you should update your local source tree structure on a regular basis.

Kernel

```
host> cd ~/Desktop/Raspberry/kernel
host> git pull --tags
```

Moitessier HAT

```
host> cd ~/Desktop/Raspberry/moitessier
host> git pull origin master
host> git submodule update
```

9.4 Compilation

Kernel

You need to ensure that the Moitessier HAT sources are cross compiled for the same kernel version, that is running on your Raspberry Pi, otherwise device driver loading will fail due to wrong module format. To get the relevant information, proceed as follows.

(1) Determine the current kernel source code version.

```
pi> zcat /usr/share/doc/raspberrypi-bootloader/changelog.Debian.gz | head
```

This command will output the version of the associated kernel sources. In this example the version is `1.20180417-1`.

```

raspberrypi-firmware (1.20180417-1) stretch; urgency=medium

  * firmware as of 5db8e4e1c63178e200d6fbea23ed4a9bf4656658

-- Serge Schneider <serge@raspberrypi.org> Tue, 17 Apr 2018 13:07:14
+0100

raspberrypi-firmware (1.20180328-1) stretch; urgency=medium

  * firmware as of ce8652e2c743f02f04cb29f23611cbf13765483b

```

(2) Go to <https://github.com/raspberrypi/linux/releases> and search for *1.20180417-1*. The associated tag is *raspberrypi-kernel_1.20180417-1*.

(3) Checkout the sources referenced by the repository tag.

```

host> cd ~/Desktop/Raspberry/kernel
host> git checkout raspberrypi-kernel_1.20180417-1

```

As far as you don't change anything on the kernel sources itself or on the kernel version used on the target system, you will only need to compile the kernel once. This step is only required to cross compile the kernel device driver of the Moitessier HAT. There is no need to copy the created kernel binary to your Raspbian system.

```

host> make distclean ARCH=arm
host> KERNEL=kernel7
host> make bcm2709_defconfig ARCH=arm
host> make -j2 ARCH=arm CROSS_COMPILE=arm-linux-gnueabi- zImage dtbs
modules

```

Moitessier HAT

The compilation/generation process is managed by the bash script *create*, located in the top level of the source directory. This script is used to compile all Moitessier HAT related sources and creates in the final step a debian package. However, you might want to build the sources step by step. In this case, use the proper scripts/Makefiles in the subdirectories. You'll get the necessary instructions in the README.md files included in the subdirectories.

If you make changes on the source code file structure or you want to use a different compiler, you might need to adopt the configuration in this script.

```

host> cd ~/Desktop/Raspberry/moitessier
host> ./create

```

9.5 Package Installation

After you've compiled the sources successfully, you can copy the created debian package to your Raspberry Pi using *scp* or similar. Use the command below with a proper IP address to access the Raspberry Pi in your network.

```

host> scp -r deploy/moitessier_*_armhf.deb pi@10.10.10.1:/home/pi/Desktop

```

Afterwards you need to install the package on your Raspberry Pi.

```

pi> sudo dpkg -i ~/Desktop/moitessier_*_armhf.deb

```

Your system will reboot and will load the device driver to access the Moitessier HAT automatically. No need to manually load the driver.

To check the installed package version, call the following.

```
pi> dpkg -s moitessier
```

To check the version of the package without installation, use the following command.

```
dpkg-deb -I package.deb
```

Compatibility with OpenPlotter

If you are using OpenPlotter copy the package to the configuration directory.

```
host> scp -r deploy/moitessier*_armhf.deb  
pi@10.10.10.1:/home/pi/.config/openplotter/tools/moitessier_hat/packages
```

The package will be listed directly within the Moitessier HAT settings window in OpenPlotter (see section 8.2). You might need to restart the OpenPlotter GUI to update the list.

9.6 Virtual Machine

For simplicity a prepared virtual machine can be used, including the relevant sources and all the tools required to create a Raspbian compatible package. It is assumed that you have the free VMware Player or VirtualBox installed on your development host machine.

Virtual machine: <http://downloads.rooco.eu/moitessier/vm.zip>

MD5 hash: http://downloads.rooco.eu/moitessier/vm_md5.txt

You might want to verify the integrity of the downloaded file. Use a utility like *Hash Check 4dots*³ to generate the MD5 hash. Compare this hash with the hash stated in *vm_md5.txt*.

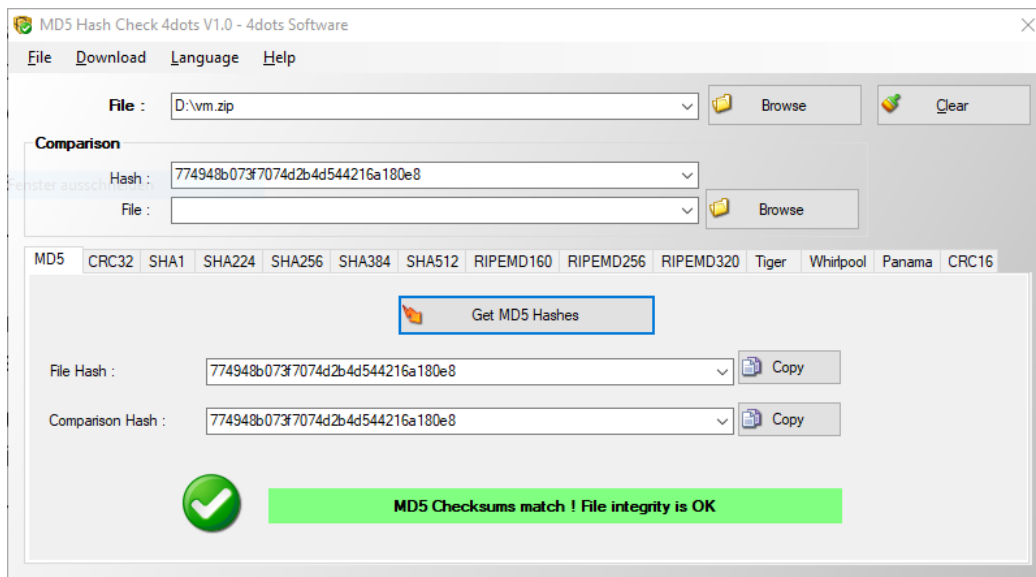


Figure 17: Integrity verification with hash utility

Unpack the virtual machine, open the *.ovf configuration file in the VMware Player / VirtualBox and start the virtual machine.

If using the VMware Player, you might get an import warning. Press the retry button to continue the import.

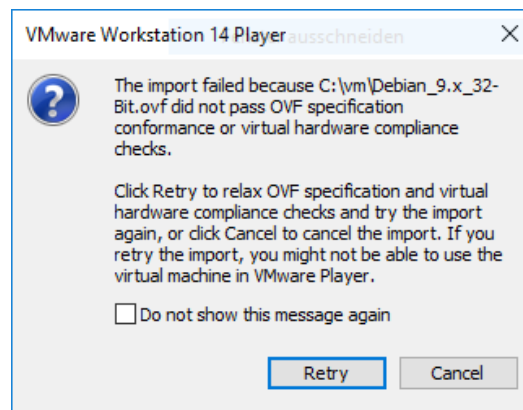


Figure 18: VMware Player, import warning

³ <https://www.4dots-software.com/md5hashchecker/>

Login credentials:

- User: development
- Password: 1234

The user *development* belongs to the group *sudo*, that enables root rights required for compilation purpose.

The relevant sources can be found on the Desktop in the folder *Raspberry*:

- *kernel*: kernel sources for the Raspberry Pi
- *moitessier*: sources related to the Moitessier HAT (kernel driver, applications, scripts, etc.)
- *toolchain*: cross toolchain used to compile the kernel and Moitessier sources

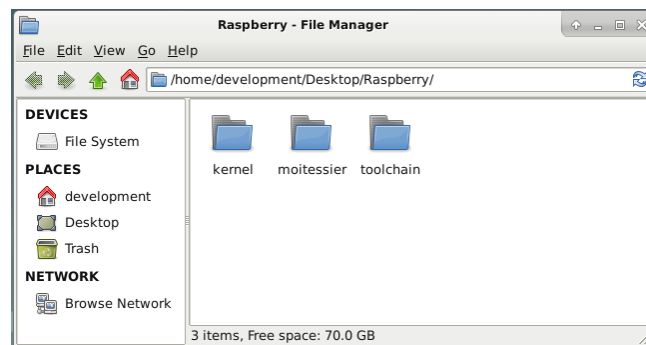


Figure 19: Source folders

To compile the sources and to install the package afterwards, proceed as described in section 9.3 to 9.5.

The virtual machine uses English as system language. However, the keyboard layout is configured for German. You might want to change this via the menu *Application* → *Settings* → *Keyboard*.

10 Communicating with the HAT



This chapter focuses on the communication with the HAT to read AIS/GNSS and sensor data. The mentioned commands will only work, if you have exclusive access to the HAT. They are intended to be used with a system without OpenPlotter. If you are using OpenPlotter, proceed as described in chapter 8.

The device driver for the Moitessier HAT is automatically loaded during system booting, if the package is installed as described in the previous section. You can communicate with the HAT using the following device files:

- `/dev/moitessier.tty`: Used to read data from the HAT.
- `/dev/moitessier.spi`: Similar to TTY, but slightly different access mode.
- `/dev/moitessier.ctrl`: Used to configure/control the HAT and to read statistic information.

10.1 Reading AIS and GNSS Data

To test if data can be read, open a terminal and use one of the following commands. If you are too far away from real AIS targets, you might want to activate the AIS simulator of the HAT (see section 10.3). However you should at least receive GNSS data.

```
pi> cat /dev/moitessier.tty
pi> cat /dev/moitessier.spi
```

If this is successful, you can open `/dev/moitessier.tty` in any navigation software, that can read data from serial inputs (e.g. OpenCPN).

Be aware, that the interface can only be opened by a single process.

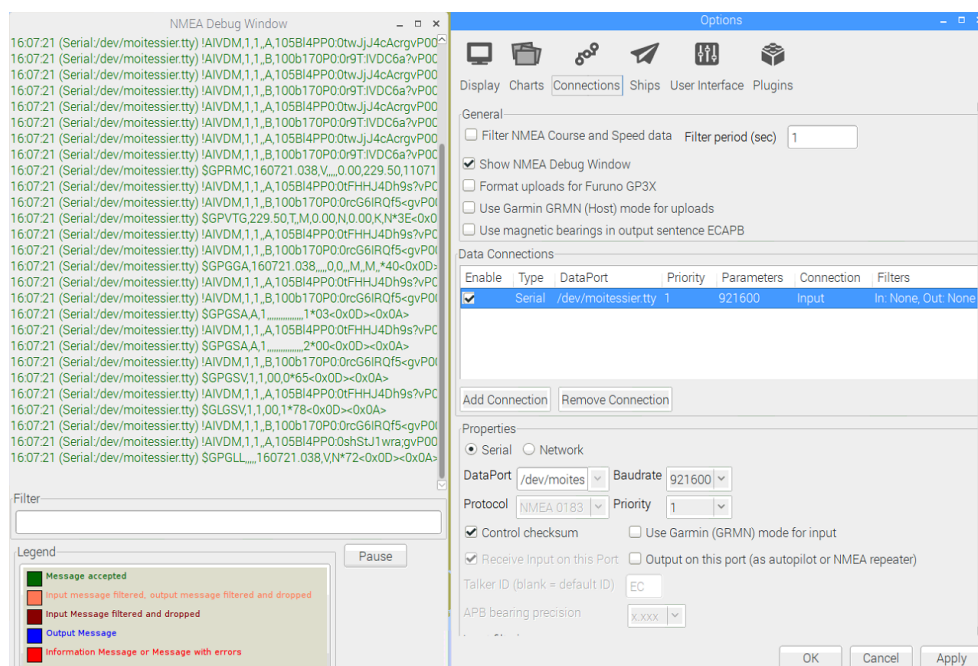


Figure 20: Using the HAT with OpenCPN

10.2 Reading Sensor Data

The sensors can be read via I²C directly by the Raspberry Pi. There are sample programs in the directory *moitessier/app/sensors* that might help to write your own applications.

To read the sensors using the sample programs, use the following commands.

```
pi> ~/moitessier/app/sensors/MPU-9250 /dev/i2c-1 1 1
pi> ~/moitessier/app/sensors/MS5607-02BA03 /dev/i2c-1 1 1
pi> ~/moitessier/app/sensors/Si7020-A20 /dev/i2c-1 1 1
```

To get the available program options, call the sample program without parameter.

You might also want to use the I²C tools available in Raspbian (package *i2c-tools*) to verify if the sensors are accessible. The command should at least return addresses 0x68 (MPU-9250) and 0x77 (MS5607-02BA03). Depending on your HAT model, you'll also get 0x40 (Si7020-A20).

```
pi> i2cdetect -y 1
```

10.3 Controlling/configuring the HAT

You'll find the program *moitessier_ctrl* in subdirectory *moitessier/app/moitessier_ctrl*. It can be used to read statistics from the HAT, to receive HAT information, to reset the HAT, to enable/disable specific features (e.g. AIS simulator), etc.

Call the program without parameters to get the available options.

```
pi> ~/moitessier/app/moitessier_ctrl/moitessier_ctrl
```

Example: To read the HAT information you might call the following.

```
pi> ~/moitessier/app/moitessier_ctrl/moitessier_ctrl /dev/moitessier.ctrl 1
```

To configure the receiver and/or enable/disable specific features you need to write a configuration file (XML) to the HAT. You can edit this file with any text editor (e.g. *nano*, *vi*). The original file includes comments and is self-explanatory.

```
pi> cd ~/moitessier/app/moitessier_ctrl
pi> ./moitessier_ctrl /dev/moitessier.ctrl 5 config.xml
```

The default configuration is available in *default_config.xml*.

AIS Simulator

The Moitessier HAT does feature an AIS simulator, that can be enabled and disabled via the configuration file. The simulator can be used to test the communication between Raspberry Pi and HAT without receiving real AIS signals. The HAT will generate two simulated AIS targets in the Mediterranean Sea.

The following XML tag snippet will enable the simulator with an interval of 100 ms and the MMSI numbers 5551122 for target 1 and 688412 for target 2.

```
<simulator>
  <enabled>1</enabled>
  <interval>100</interval>
  <mmsi>
    <id>5551122</id>
    <id>688412</id>
  </mmsi>
</simulator>
```



The configuration is not permanently stored in the HAT, so you will need to reconfigure the HAT whenever you reset the HAT or the Raspberry Pi.

11 Status LEDs

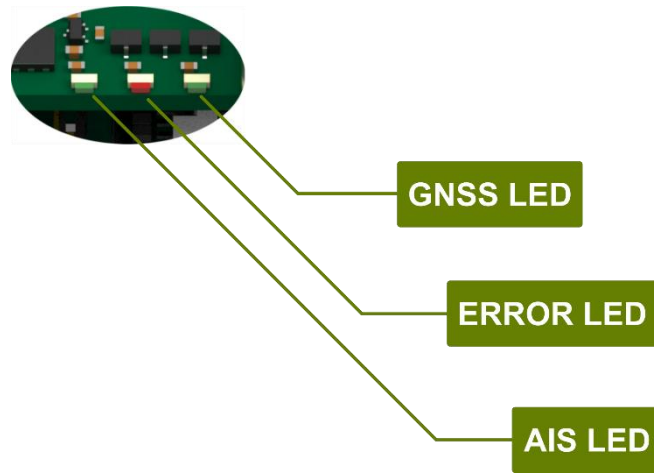













Figure 21: Status LEDs

Meaning of LED Sequence

-  LED switched off
-  LED switched on
-  LED flashing at variable/inconsistent frequency
-  LED flashing at consistent frequency

LED Patterns

Status LED	Color	Pattern	Meaning
ERROR	red		No errors occurred
		 <i>Duration: as long as the error exists</i>	Internal buffer overflow. The data is processed too slowly by the Raspberry Pi.
		 <i>Duration: until reset</i>	Error occurred on device self-test
AIS	green		No AIS data available
			AIS data being received
GNSS	green		No GNSS fix
			GNSS fix

Error Output

The *ERROR* LED indicates the following error types:

- *Minor/temporary error*: Every NMEA source (AIS, GNSS) and every NMEA output (SPI interface to Raspberry Pi) is assigned a memory area in the microcontroller in the form of a buffer. The data of multiple inputs is written to the SPI buffer. If data is written to this buffer quicker than it is read by the Raspberry Pi, the buffer will be full after a certain time due to its memory restrictions. In such a case, no more data can be written and a buffer overflow occurs. As soon as the buffer is read, the device continues to process data.
- *Serious (system) errors*: The device is running a self-test after each reset. If an error occurs during hardware initialization, the *ERROR* LED flashes in a constant pattern. The device can no longer be used in this case. If a restart does not resolve the problem, please contact the Rooco support team.

You can read the system errors by reading the device information.

```
pi> ~/moitessier/app/moitessier_ctrl/moitessier_ctrl /dev/moitessier.ctrl 1
```

The error is coded as bit pattern:

- Bit 0: GNSS failed
- Bit 1: AIS receiver 1 failed
- Bit 2: AIS receiver 2 failed
- Bit 3: SPI host interface failed
- Bit 4: UART host interface failed

e.g. system errors = 0x00000011 → GNSS and UART host interface failed

12 Receiving NMEA Data over Wi-Fi

You can use your OpenPlotter powered Raspberry Pi as headless data distribution system. Connect your Wi-Fi enabled device (smartphone, tablet, etc.) to the Raspberry Pi and configure the following IP settings in your navigation app:

- Source IP: 10.10.10.1
- Port: 10110
- Protocol: TCP

The following subsections will show the configuration of some popular AIS navigation tools/apps.

12.1 iSailor

- Start iSailor
- Open the sensor settings **1 2**
- Select *NMEA* as primary position source **3**
- Open the connection settings **4** and add a new connection **5**
- Select *TCP* as protocol, *10.10.10.1* as the address and *10110* as the port **6**

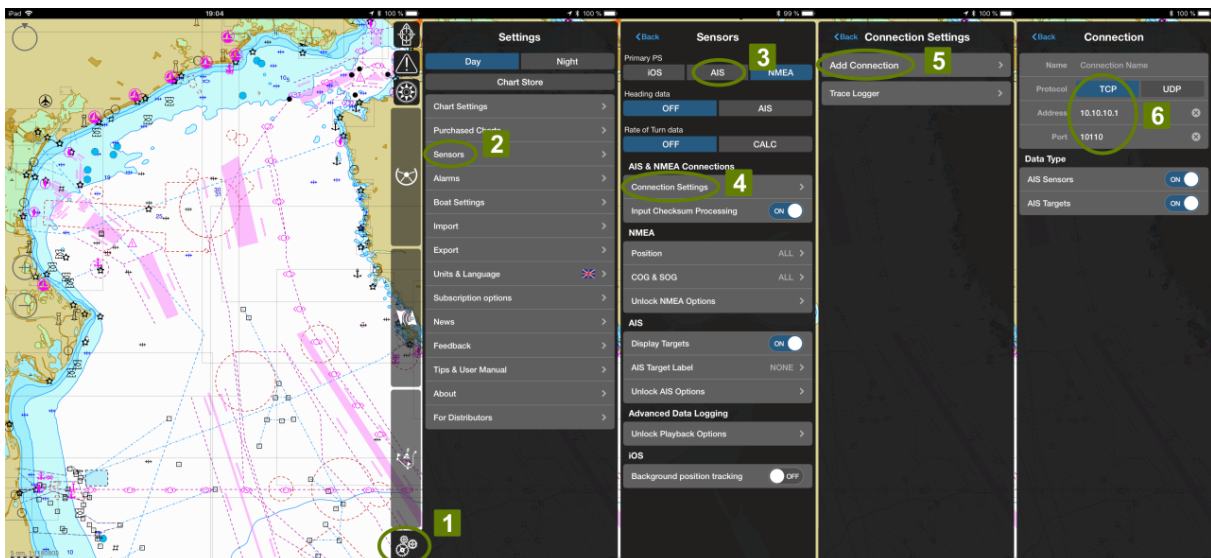


Figure 22: iSailor – TCP configuration

12.2 NV Chart

- Start NV Chart
- Open the settings **1**
- Switch to the *GPS/AIS* tab **2**
- Select *NMEA GPS (RMC)* as position source **3**
- Select *Wifi/TCP* as the NMEA source **4**, *10.10.10.1* as the hostname / IP address and *10110* as the port **5**
- Accept the settings with *Done* **6**

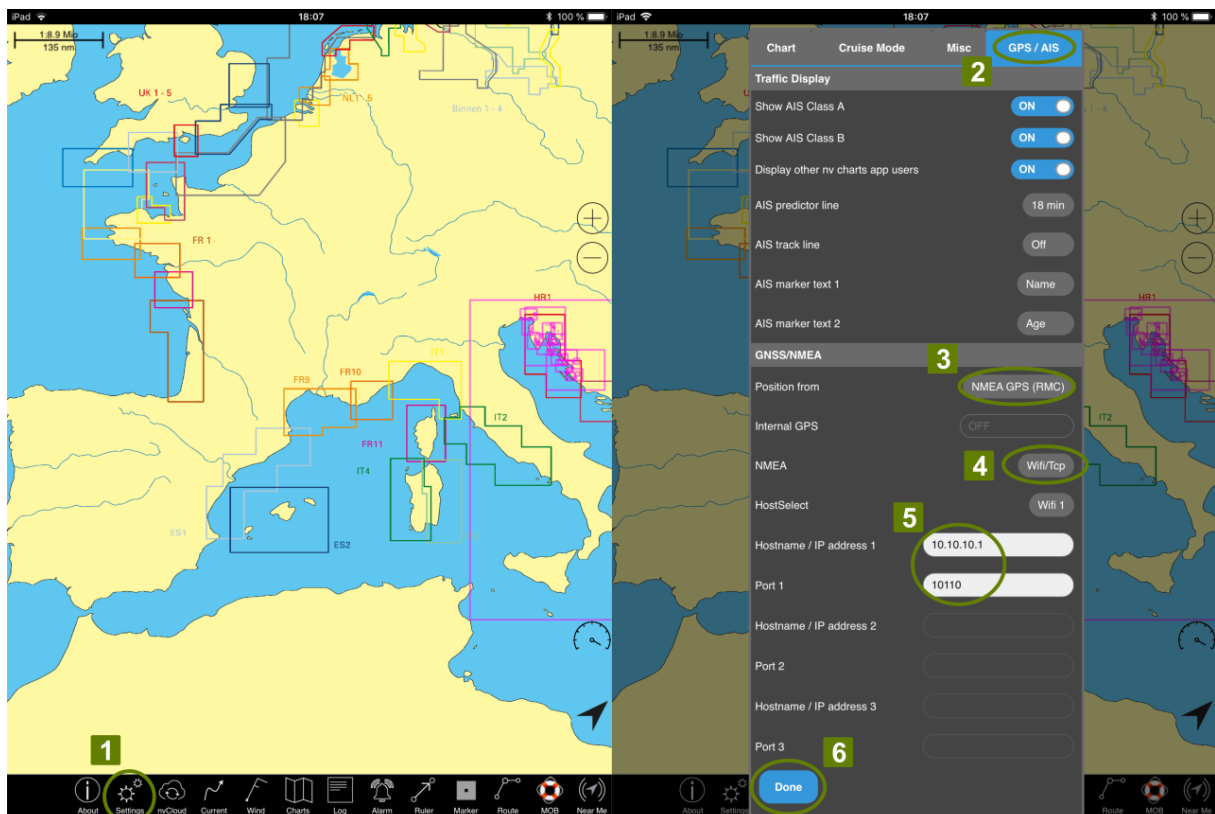


Figure 23: NV Chart App – TCP configuration

12.3 iNavX

- Start iNavX
- Switch to the settings **1 2**
- Open the *TCP/IP NMEA Client settings* **3**
- Select *10.10.10.1* as host IP address and *10110* as the data port **4**
- Open the interface by clicking the *link* button **5**. You will NMEA messages in the output debug window.

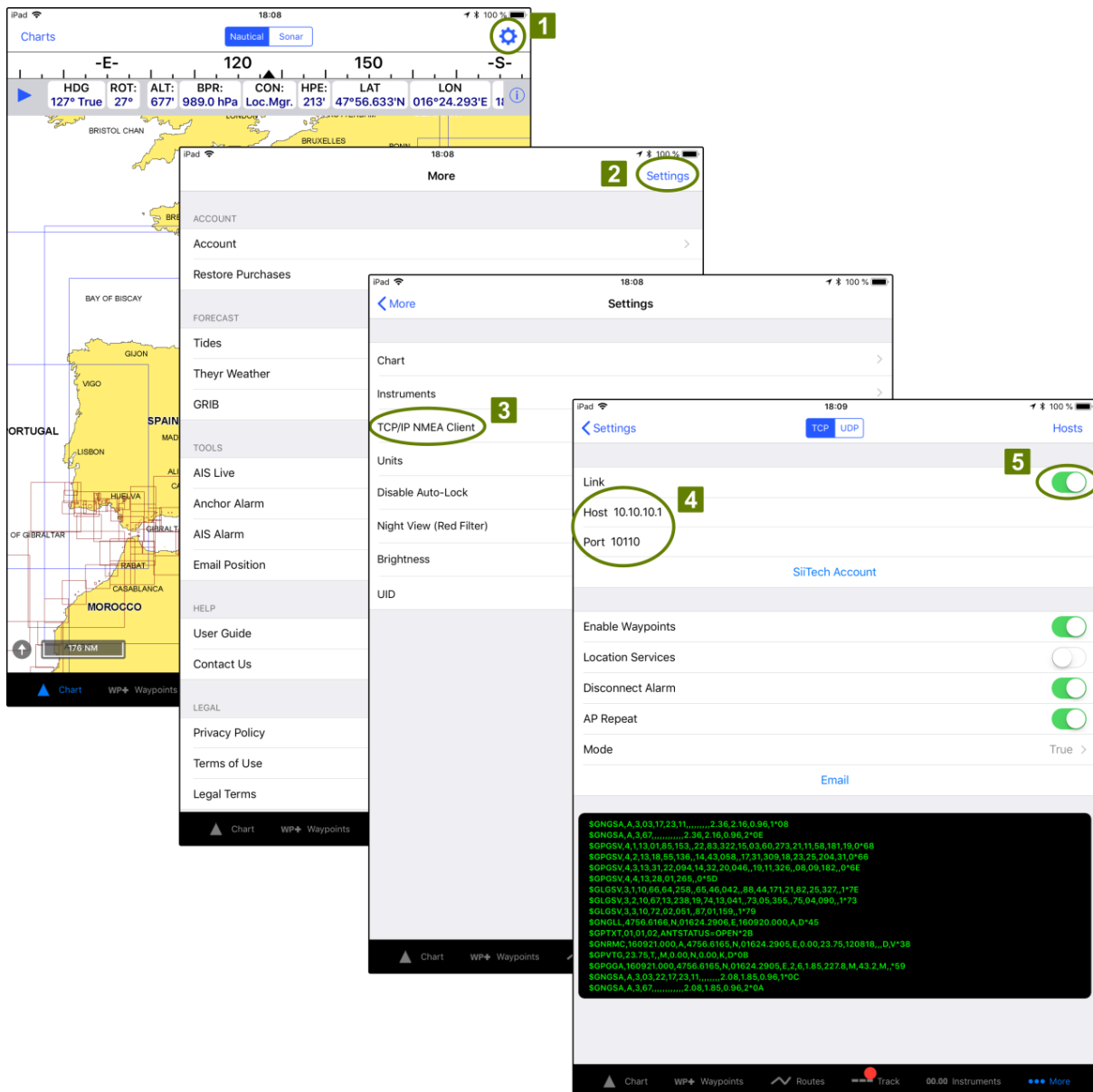


Figure 24: iNavX – TCP configuration

13 Contact and Support Information

csoft – Web and IT Solutions

Wiener Straße 2

8665 Langenwang

AUSTRIA

Phone: +43 (0)3854 25701

Fax: +43 (0)3854 25701 - 20

E-mail: info@csoft-it.at
support@rooco.eu (support requests only)

Web: <https://www.rooco.eu>

Please do not return any device without prior consultation of our support team. Often a problem can be solved quickly on the phone or via e-mail.

At least the following information is required for any support request:

- Telephone number to call back (if required)
- Purchase date including invoice number if known
- Detailed fault description
- Serial number
- Hardware number (P77xxxx)
- Antenna used (if relevant)
- Operating system used
- Navigation software used (if relevant)

14 Part Numbering Scheme

The Moitessier HAT features some options, that extend the functionality of the basic model and need to be specified when ordering.

Example: **PE77001Exx – DBG / ... / 5V**

Device Type

PE77001 (Moitessier Navigation HAT)

Revision

Exx = latest version
 E01 = version 1
 E02 = version 2
 etc.

Options

DBG = debug header available
 IO = IO headers available
 IPEX = IPEX connectors instead of SMA/BNC
 M55 = MPU-9255 instead of MPU-9250 (*note: MPU-9255 is obsolete, availability limited and not guaranteed*)
 UART = Raspberry Pi and HAT UART available on header
 SENS = additional temperature and humidity sensor
 256K = HAT microcontroller featuring 256kByte flash memory instead of 128K
 5V = HAT is powered by 5V instead of 3.3V

Options need to be separated by / and arranged in alphabetical order.

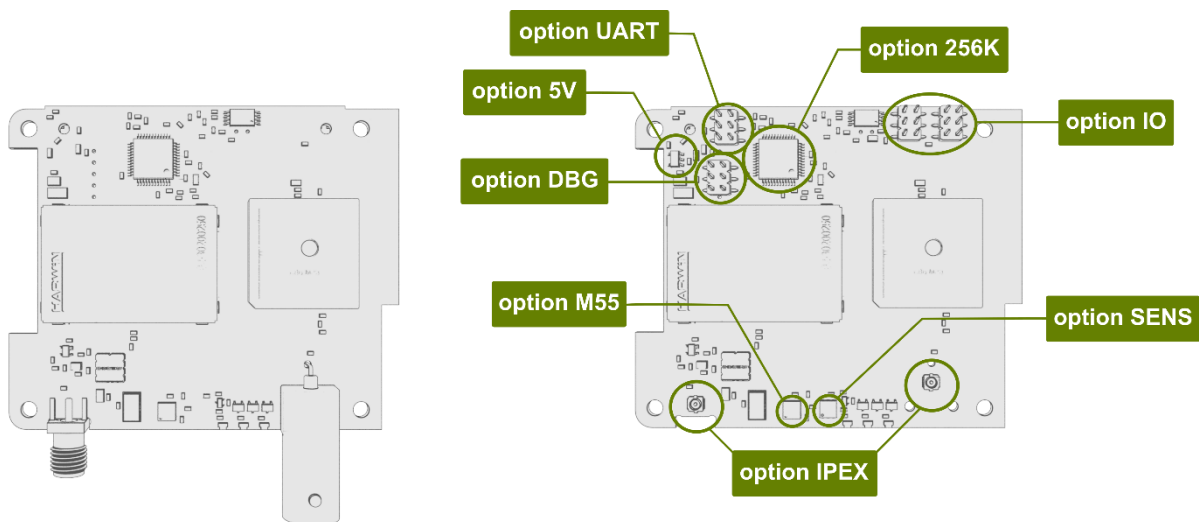


Figure 25: Basic model (left, PE77001Exx), fully equipped model (right, PE77001Exx-DBG/IO/IPEX/M55/UART/SENS/256K/5V)

15 Technical Specification

Parameter	Meaning	Min.	Max.	Unit
General				
Supply voltage		3.135	3.465	V
Current consumption	Excl. HAT headers		185	mA
Operating temperature range		-20	65	°C
Dimensions	Excl. connectors		69x57	mm
Weight			39	g
AIS Receiver				
Interface	Seen from Raspberry Pi, receiver indirectly accessible through HAT microcontroller	SPI		
Input frequency	Channel spacing = 50 kHz	161.975	162.025	MHz
Input sensitivity	20 % PER, nominal input frequency	-112		dBm
Output data rate	Net data rate		9600	bits/s
	Packet interval <i>SPI communication interval between HAT and Raspberry Pi depends upon driver configuration</i>		37.5	Hz
BT			0.5	
GNSS Receiver				
Interface	Seen from Raspberry Pi, receiver indirectly accessible through HAT microcontroller	SPI		
Data rate	Net data rate		9600	bits/s
	Packet interval <i>SPI communication interval between HAT and Raspberry Pi depends upon driver configuration</i>		1	Hz
Channels	Acquisition		99	
	Tracking		33	

Supported satellite systems		GPS, GLONASS, QZSS		
Sensitivity	Acquisition		-148	dBm
	Tracking		-165	dBm
Dynamic performance	Altitude		18000	m
	Speed		515	m/s
	Acceleration		4	G
Cold start			35	s
Warm start			30	s
Horizontal position accuracy		2.5		m
Humidity and Temperature Sensor (Si7020-A20) - optional				
Interface	Directly accessible by Raspberry Pi	I ² C		
Slave address	7 bit	0x40		
Resolution	ADC, relative humidity		12	bits
	ADC, temperature		14	bits
Conversion time	12-bit relative humidity (RH)		12	ms
	14-bit temperature		10.8	ms
Operating range	Non-condensing	0	100	% RH
Accuracy	0-80% RH, T _A =30°C		±4	% RH
	> 80% RH, T _A =30°C		±6.5	% RH
	-10°C < t _A < 85°C		±0.4	°C
	-40°C < t _A < -10°C		±0.9	°C
	-85°C < t _A < -125°C		±1.1	°C
Response time	RH sensor, T _{63%} , 1 m/s airflow, without cover		17	s
	Temperature sensor		7	s
Long term stability	RH sensor, typ. value		≤ 0.25	%RH/year
	Temperature sensor, typ. value		≤ 0.01	°C/year
Barometric Pressure Sensor (MS5607-02BA03)				
Interface	Directly accessible by Raspberry Pi	I ² C		
Slave address	7 bit	0x77		
Operating range		10	1200	mbar

Conversion time	Oversampling Ratio: 4096	7.4	9.04	ms
	Oversampling Ratio: 256	0.48	0.6	ms
Resolution	ADC		24	bits
	Oversampling Ratio: 256 / 512 / 1024 / 2048 / 4096	0.13/0.084/0.054/0.036/0.024		mbar
Accuracy	T _A =25°C, 750 mbar		±1.5	mbar
Response time	Oversampling Ratio: 256 / 512 / 1024 / 2048 / 4096	0.5/1.1/2.1/4.1/8.22		ms
Long term stability	Typ. value		±1.5	mbar/year
Motion Tracking Sensor (MPU-9250, gyroscope + accelerometer + magnetometer)				
Interface	Directly accessible by Raspberry Pi	I ² C		
Slave address	7 bit	0x68		
Gyroscope				
Full-scale range	FS_SEL=0, typ. value		±250	%s
	FS_SEL=1, typ. value		±500	%s
	FS_SEL=2, typ. value		±1000	%s
	FS_SEL=3, typ. value		±2000	%s
Sensitivity scale factor	FS_SEL=0, typ. value		131	LSB/(°/s)
	FS_SEL=1, typ. value		65.5	LSB/(°/s)
	FS_SEL=2, typ. value		32.8	LSB/(°/s)
	FS_SEL=3, typ. value		16.4	LSB/(°/s)
Resolution	ADC		16	bits
Nonlinearity	Best fit straight line, T _A =25°C, typ. value		±0.1	%
Output data rate		4	8000	Hz
Initial zero tolerance	T _A =25°C		±5	%s
Accelerometer				
Full-scale range	AFS_SEL=0, typ. value		±2	g
	AFS_SEL=1, typ. value		±4	g
	AFS_SEL=2, typ. value		±8	g
	AFS_SEL=3, typ. value		±16	g
Sensitivity scale factor	AFS_SEL=0, typ. value		16384	LSB/g
	AFS_SEL=1, typ. value		8192	LSB/g
	AFS_SEL=2, typ. value		4096	LSB/g
	AFS_SEL=3, typ. value		2048	LSB/g

Resolution	ADC		16	bits
Initial tolerance	Typ. value		±3	%
Nonlinearity	Best fit straight line, T _A =25°C, typ. value		±0.5	%
Output data rate		0.24	4000	Hz
Magnetometer				
Full-scale range	Typ. value		±4800	μT
Resolution	ADC		14	bits
Sensitivity scale factor	Typ. value		0.6	μT/LSB
Initial calibration tolerance	Typ. value		±500	LSB