Documentation
Documentación

v0.9.0.

sailoog.com

Table of Contents

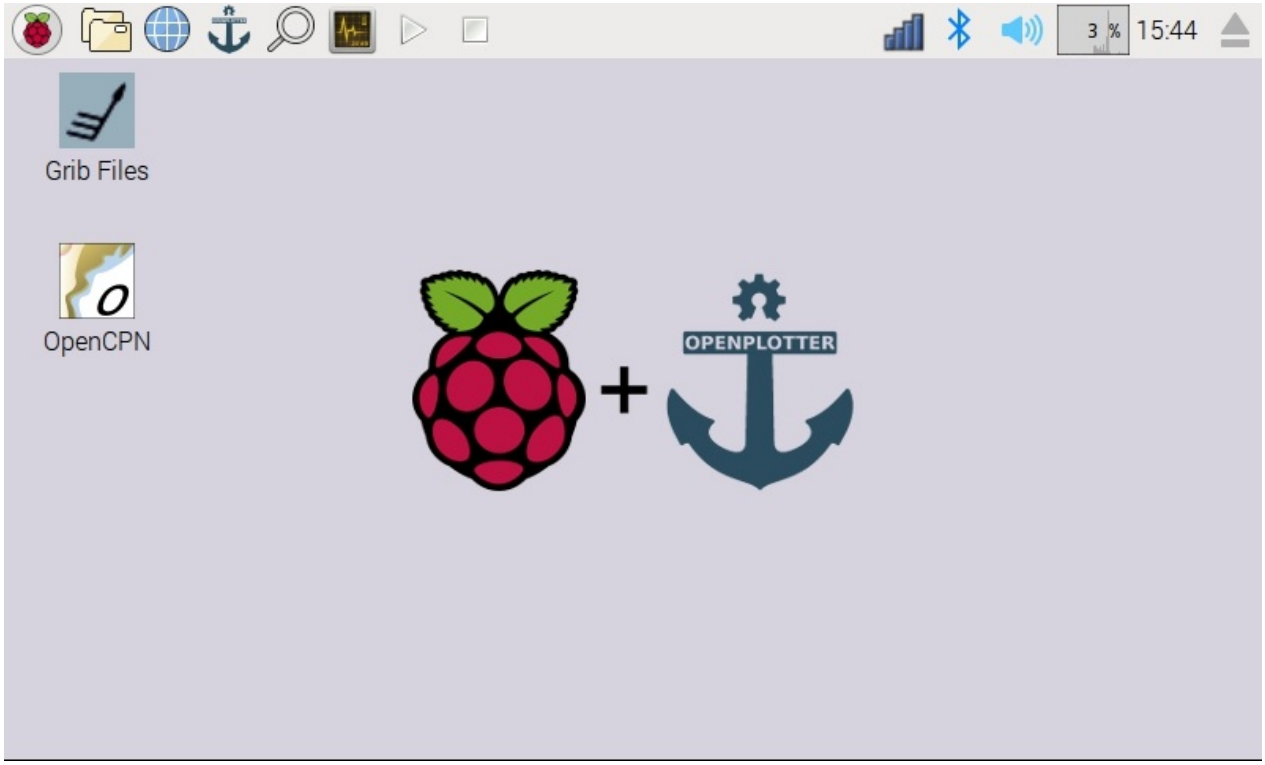
What is OpenPlotter?

There are people who buy boats but there are also people who build them, why not build your own electronics too? 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. Do it yourself.

Features

- **Chartplotter**. With OpenCPN, a navigation software with useful plugins.
- **Weather Forecast**. Download and visualize GRIB files with zyGrib.
- **NMEA 0183 Multiplexer**. Multiplex and filter data inputs from any number of serial and network interfaces. Send and filter to any number of outputs.
- **SignalK**. OpenPlotter uses SignalK as the base protocol to interact between different networks. Input data has to be converted into SignalK. SignalK data can be converted to NMEA networks.
- **Diagnostic**. Checks the data traffic of NMEA0183 input and output. On Signal K it is a kind of data explorer.
- **WiFi Access Point, Ethernet, GSM**. Share data (NMEA 0183, Signal K, remote desktop, Internet connection, gofree) with laptops, tablets and phones on board. Connect to internet on port through the same device.
- **Remote Desktop**. Access to OpenPlotter desktop from the cockpit through your mobile devices.
- **Headless**. Easy start without monitor.
- **SDR-AIS**. Receive and decode AIS with cheap DVB-T dongles. Calibration tools Included.
- **Electronic Compass and Heel**. Read magnetic heading and heel angle from an IMU sensor. Tilt compensated. Calibration tools Included.
- **Barograph, Thermograph and Hygrograph**. From pressure, temperature and humidity sensors.
- **Multiple temperature sensors**. Get data from coolant engine, exhaust, fridge, sea...
- **Special Sensors**. Detect opening doors, windows, tank level, human body motion...
- **Wireless Sensors**. Detect temperature, humidity, ...
- **Analog Sensors**. To read voltage, amperage, resistance. Shows Battery status, power consumption, tank level...
- **Magnetic Variation**. Calculate magnetic variation for date and position.
- **True Heading**. Calculate true heading from magnetic variation and magnetic heading.
- **True Wind**. Calculate true wind from apparent wind and either speed through water (speed log) or speed over ground (GPS).
- **Rate Of Turn**. Calculate the rate the ship is turning.
- **Remote Monitoring**. Publish data on Twitter or send it by email.

- **Actions System**. Compare a custom value with any data flowing through your system and use it as a trigger to run multiple predefined actions.
- **Custom Switches**. Connect external switches and link them with actions.
- **Handle External Devices**. Relays, LEDs, buzzers ...
- **System Time Tools**. Set the system time from NMEA data and set the time zone easily.
- **Tools**. Integration of self programmed modules and advanced feature modules.
- **Startup Programs**. Select some program parameters to automatically launch at start.



v0.9.0

FAQ

Q: How to connect from smartphone to SignalK Instrumentpanel?

A: Be sure you are connected to the openplotter access point. Open your browser. Type in 10.10.10.1:3000. Click on Instrumnetpanel. On the right side of the Instrumentpanel is a sign select it to open the menubar. Select *Connect* and enter in *New SignalK Server* 10.10.10.1:3000

Q: Can I create my own SignalK names?

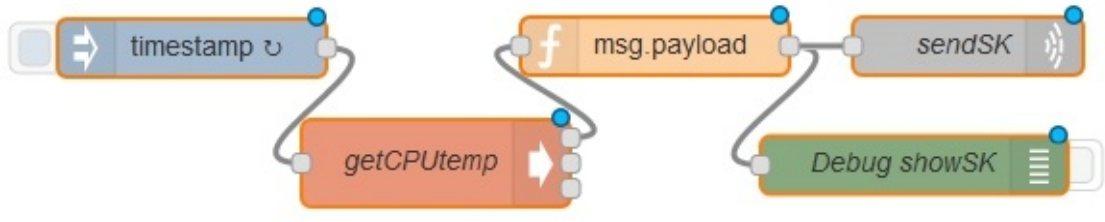
A: You can but it is recommended that you don't (to develop more standard names look for specifications in Github SignalK). If you want to have an overview of already existing SignalK names open the *diagnostic SignalK input* window and click on *Unit Setting*. If there is a star in the SignalK name you should put in an easy understandable name.

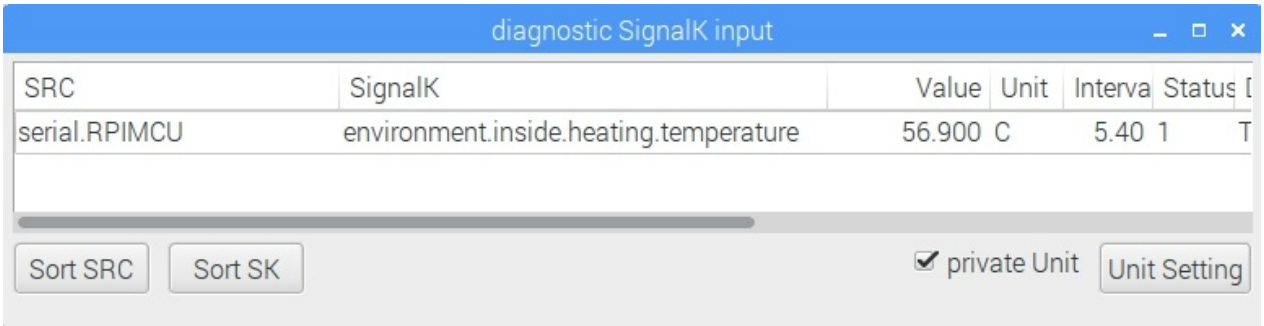
Q: Is there also a chance to send values to the SignalK server of OpenPlotter?

A: Yes you can send values to the server when you use the minimum SignalK JSON format. For example:

```
{
    "updates": [{
            "source": {
                "type": "ARMTEMP",
                "src": "RPIMCU"
            },
            "values": [{
                    "path": "environment.inside.heating.temperature",
                    "value": "323.15"
                }
            ]
        }
    ]
}
```

In this example we want to send the RPI cpu temperature to the SignalK Server.





We diidn't find a better SignalK name than environment.inside.heating.temperature.

You can import the code to test it on your own.

```
[{"id":"f5fa759f.be894","type":"inject","z":"4deb498f.fbcfd8","name":"","topic":"","payload":"","payloadType":"date",
"repeat":"10","crontab":"","once":true,"x":131,"y":127,"wires":[["adb4b4a8.9c10c8"]]},{"id":"adb4b4a8.9c10c8","type":
"exec","z":"4deb498f.fbcfd8","command":"vcgencmd","addpay":false,"append":"measure_temp","useSpawn":"","timer":"","na
me":"getCPUtemp","x":258,"y":186.5,"wires":[["2705f134.792f86"],[],[]]},{"id":"7b5088a.c5f0af8","type":"debug","z":"4
deb498f.fbcfd8","name":"Debug showSK","active":false,"console":"false","complete":"payload","x":507,"y":187.5,"wires"
:[]},{"id":"2705f134.792f86","type":"function","z":"4deb498f.fbcfd8","name":"msg.payload","func":"cpu_temp = parseFlo
at(msg.payload.replace(\"temp=\",\"\").replace(\"'C\\n\",\"\"));\ncpu_temp = cpu_temp + 273.15\nmsg.payload = '{\\\"u
pdates\\\": [{\\\"source\\\": {\\\"type\\\": \\\"ARMTEMP\\\",\\\"src\\\" : \\\"RPIMCU\\\"},\\\"values\\\":[{\\\"path\
\\": \\\"environment.inside.heating.temperature\\\",\\\"value\\\":'+cpu_temp+'}]}]}\\n';\nreturn msg;","outputs":1,"n
oerr":0,"x":372,"y":126,"wires":[["7b5088a.c5f0af8","c17f4c75.737088"]]},{"id":"c17f4c75.737088","type":"udp out","z"
:"4deb498f.fbcfd8","name":"sendSK","addr":"localhost","iface":"","port":"55559","ipv":"udp4","outport":"","base64":fa
lse,"multicast":"false","x":535,"y":126,"wires":[]}]
```

How to collaborate

> Everything takes time, money, and monkeys. You need a lot from any two groups, and a little from the third. An increase in any one reduces the requirement for the other two. Change occurs when one of those three change.

Moe's Law, Navigatrix project.

Time

Get a Raspberry Pi and the required elements, download OpenPlotter RPI and test and test and test ...

Report bugs and request new features here:

www.sailoog.com/en/blog/openplotter-community

Spread the word among your friends in ports and forums.

Money

This project is financed by selling related and tested products or by voluntary contributions.

Web shop: www.sailoog.com/shop

Donations: www.sailoog.com/openplotter

If you want to make a donation but need a receipt or invoice, buy our *OpenPlotter RPI microSD Sponsoring Edition*:

www.sailoog.com/product/openplotter-rpi-microsd-sponsoring-edition

Monkeys

> Men wanted for hazardous journey. Low wages, bitter cold, long hours of complete darkness. Safe return doubtful. Honour and recognition in event of success.

Ernest Shackleton

If you have python skills, pull your commits to the github repository:

https://github.com/sailoog/openplotter

If you have electronics skills, make your suggestions here:

www.sailoog.com/en/blog/openplotter-community

If you have English language skills, helps us with documentation. Follow the steps in the next chapter Documentation.

Documentation

- Create an account in Github: github.com/join
- Send your Github account user name to www.sailoog.com/contact and wait to get permissions to edit the repository.
- Once you have permissions, login to **Gitbook** with your Github account: www.gitbook.com/login
- Go to: www.gitbook.com/book/sailoog/openplotter-documentation and click on the *Edit* button upper right.

Writing

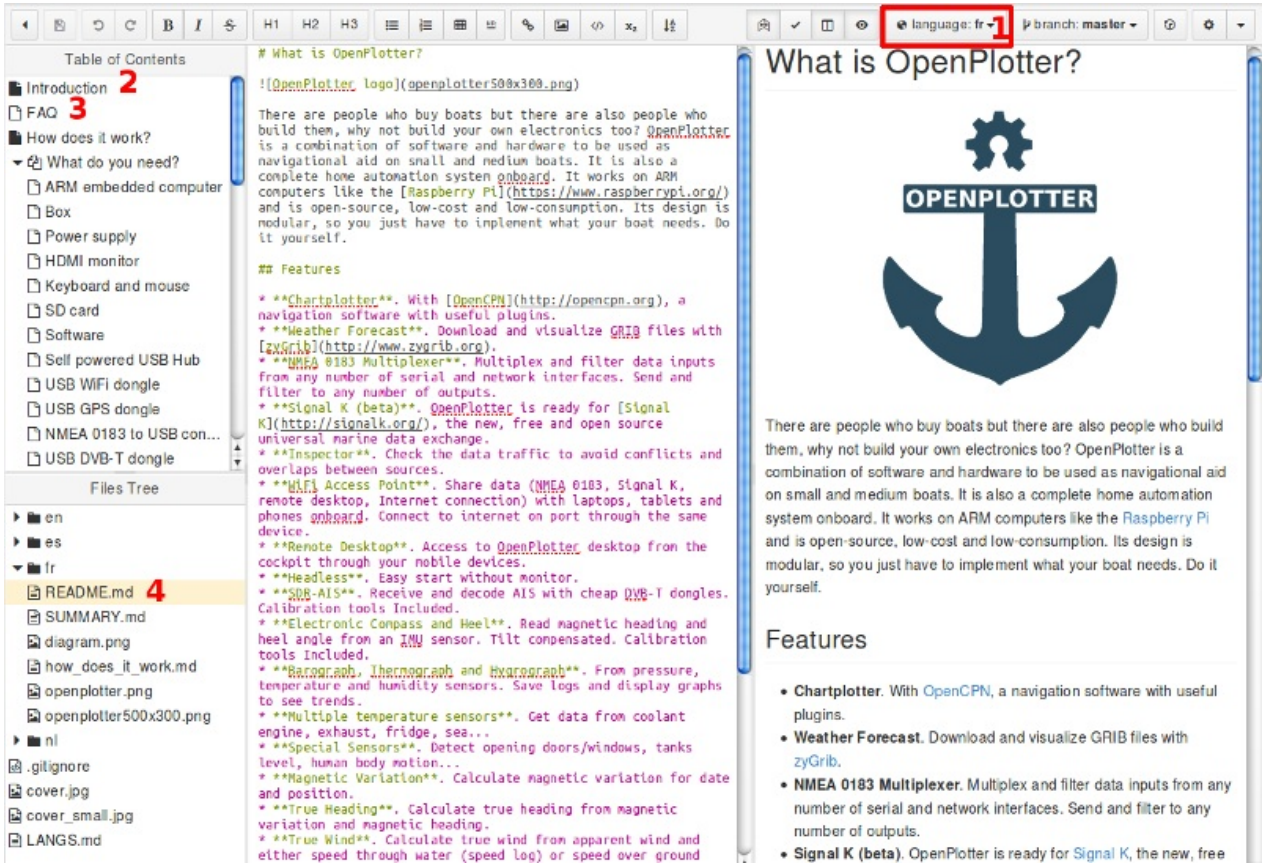
We write the documentation source in English to make easier translating to other languages. If you want to write a chapter, please notify to www.sailoog.com/contact for coordination. Please do not modify the index or write a chapter without notifying.

Once the new chapter is done in English, please copy the new text to the rest of languages to be ready to translate. If you are not creating a new chapter but editing an existing one, include a note at the beginning of the rest of languages “This translation needs to be updated”.

To include images in the rest of languages from English folder, the path is `../en/xxx.png`

If you want to add images of wiring and connections consider using fritzing application.

Translating



1. Select the language you want to edit.
2. Chapters in black are ready to be translated. You can translate chapters titles too.
3. Do not edit chapters in white.
4. Do not modify files names (xxx.md xxx.png xxx.jpg ...) because they are referenced in the text.

How does it work?

OpenPlotter can collect data from different sources:

- Sensors and devices connected to GPIO port.
- Serial devices connected to USB port.
- Other devices connected to USB port
- Any computer or portable device connected to the same network.

Most of these sources directly send data in the maritime format called NMEA 0183 or NMEA 2000. The problem is that NMEA 0183 can't communicate with NMEA 2000. So we use the free maritime communication SignalK. Both NMEA formats can communicate to SignalK.

Other devices, like SDR AIS, ADC, GPIO or some sensors, need to be processed by OpenPlotter to convert raw data to NMEA or SignalK.

All these sources are combined in data streams which can be sent to SignalK, because this is openplotter base protocol since 0.9.0:

The soft- and hardware needs different data streams

1. the good old **NMEA 0183**
2. the CAN-BUS network (standard bus-system in cars) with the special **NMEA 2000** protocol
3. **SignalK** browser optimized protocol
4. trigger\/action system

NMEA 0183 is used for:

- most pc, tablets and mobile software at the moment as the Internal chartplotter (OpenCPN).
- missing sentences can be converted from SignalK to NMEA 0183 with Openplotter tools.

NMEA 2000 is used in:

- most modern boats (plotter, engine, ...)
- very little software.

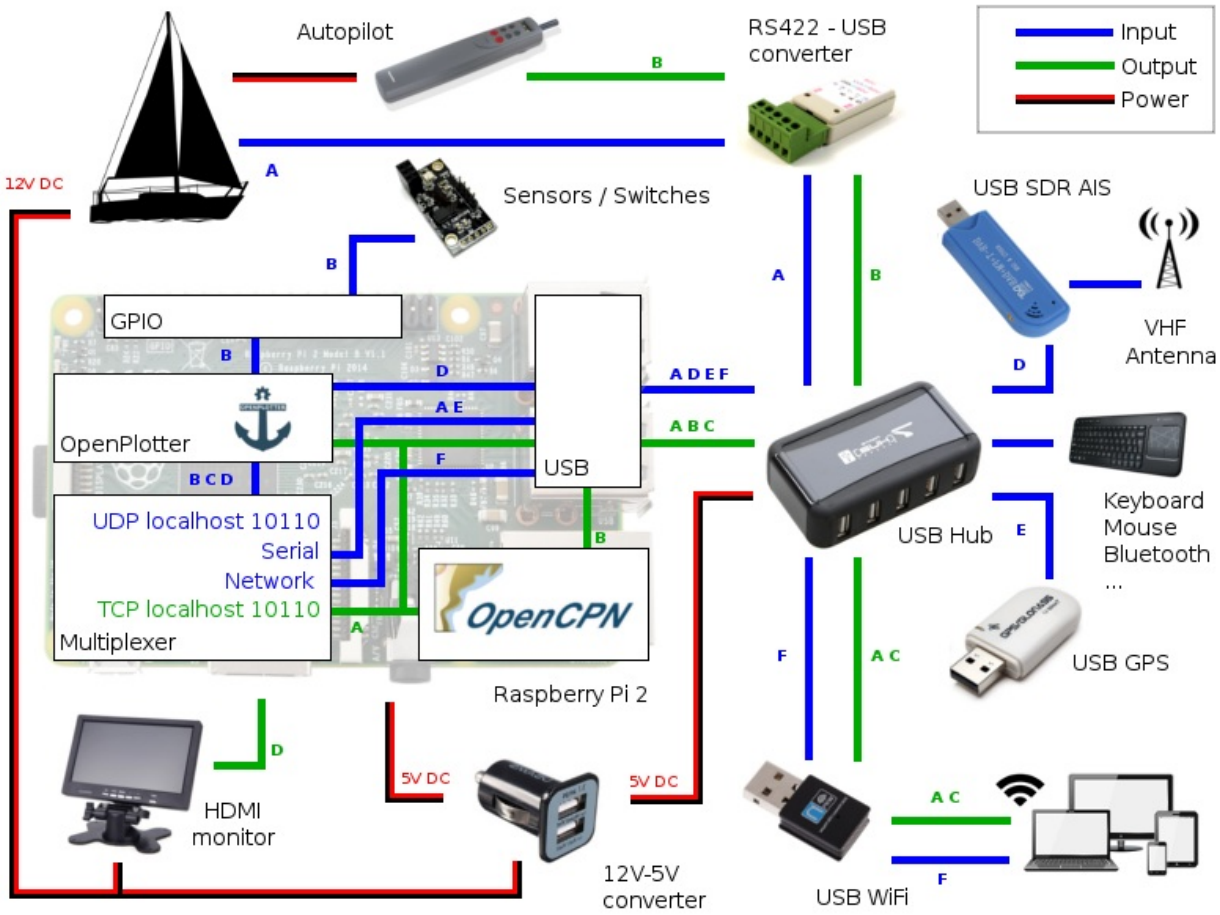
SignalK is used for:

- future software as it is optized for internet browser
- Virtual Instrument Panel
- has a readable protocol where additions can be done
- can receive NMEA 2000, NMEA 0183 and SignalK

trigger\/action system for:

- warning, information, energie saving, automation, ...
- can use twitter, e-mail, mqtt

Through the chapters of this manual we will see how to do this.



What do you need?

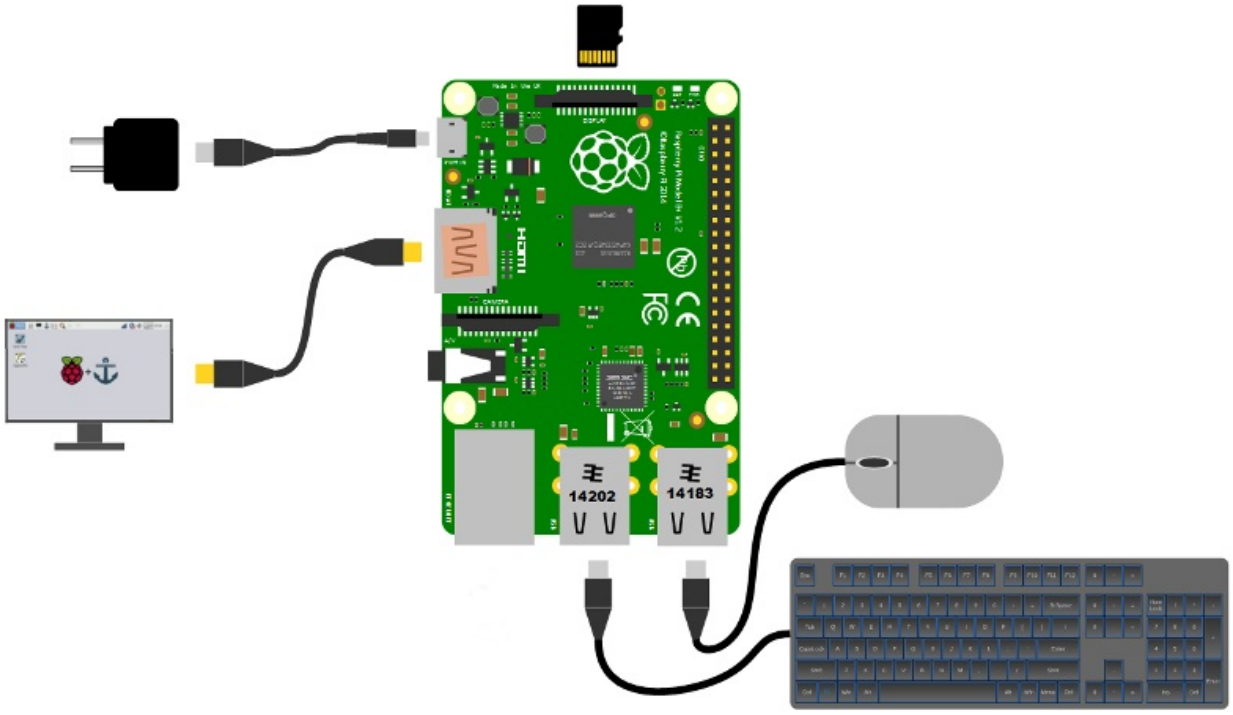
You will need the basic parts and some optional parts. It will depend on what kind of data you want to collect, processor display and what kind of equipment your boat already has.

Required items

You need at least these items to run *the software* and you have two options: either with monitor or without monitor (headless).

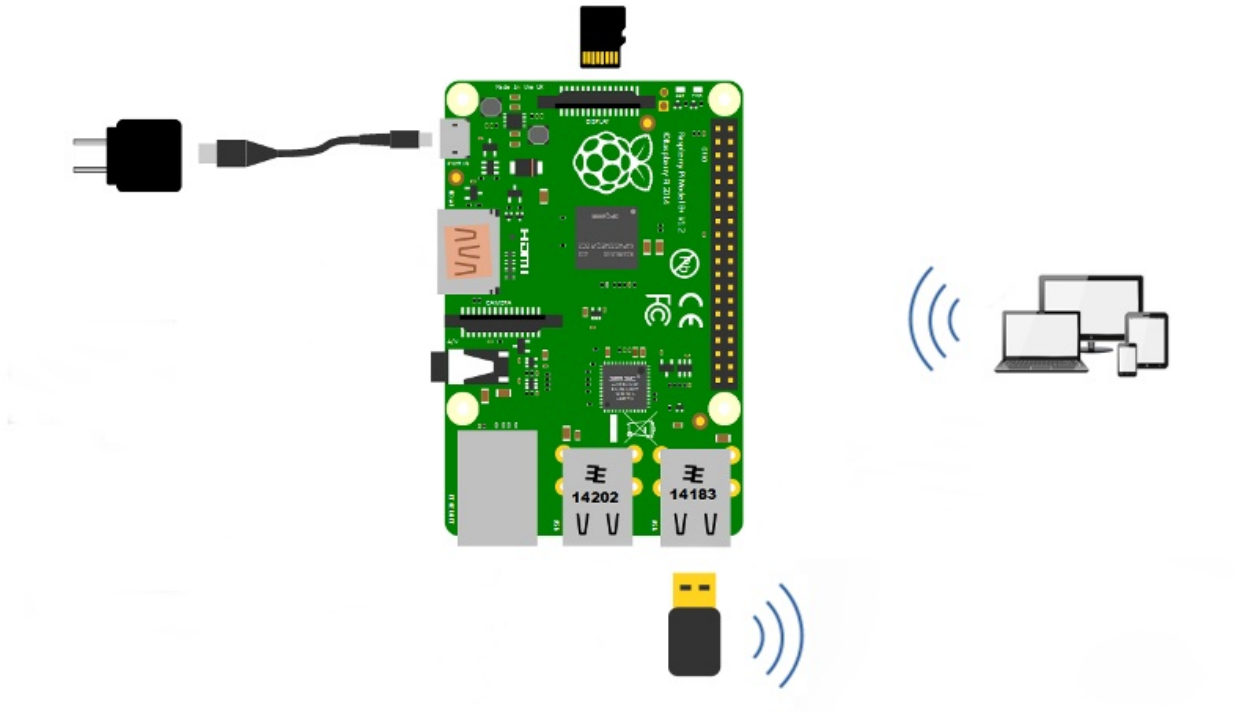
Monitor option

- ARM embedded computer (Raspberry Pi 2 or Raspberry Pi 3)
- Box
- Power supply
- HDMI monitor
- Keyboard and mouse
- SD card
- OpenPlotter RPI (*the software*)



Headless option

- ARM embedded computer (Raspberry Pi)
- Box
- Power supply
- USB WiFi dongle
- SD card
- OpenPlotter RPI (*the software*)
- Any laptop tablet or smartphone

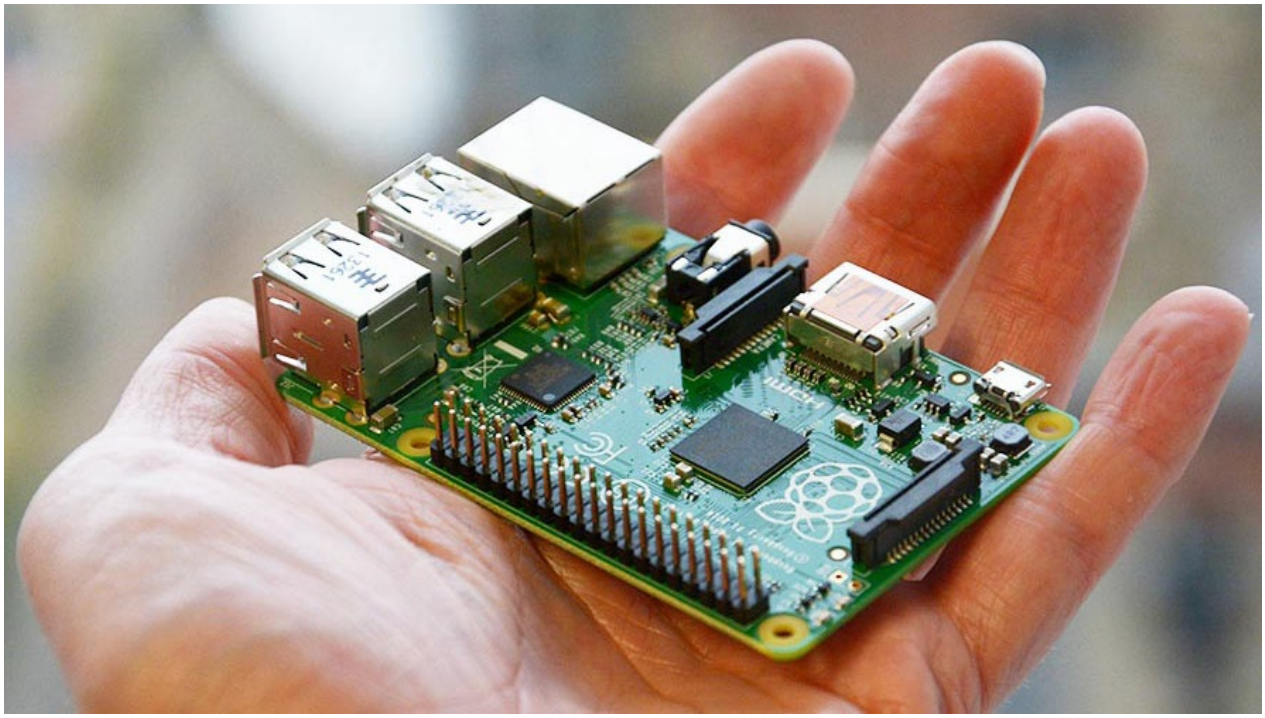


Optional items

Devices to communicate with boat and sensors to collect data from environment.

- Self powered USB Hub
- USB WiFi dongle
- USB GPS dongle
- NMEA 0183 to USB converter
- special Can-Bus to USB converter (for NMEA 2000)
- USB DVB-T dongle (AIS reception)
- IMU sensor
- Pressure\/Temperature sensor
- Humidity\/Temperature sensor
- One wire temperature sensor
- Analog Digital Converter
- PIR motion sensor
- Common switch
- Door switch
- Float switch
- Relay
- LED
- Buzzer

ARM embedded computer



We recommend the popular Raspberry Pi 2 because it fits the requeriments: open-source, low-cost, low-consumption and has a huge community of developers.

We are developing **OpenPlotter RPI**, a special operating system for Raspberry Pi based on Linux Raspbian distribution. When development is done we will migrate to other ARM boards.

You can use OpenPlotter with Raspberry Pi model Zero and model 1 but they have poor performance running OpenCPN. These models are perfect for headless systems.

Box

There are a lot of box models to protect the Raspberry Pi board. **We are working on a waterproof enclosure.**

Power supply

The Raspberry Pi is powered by a 5V USB power supply with a micro USB connector (like most standard mobile phone chargers). Exactly how much current (Amp) the Raspberry Pi requires is dependent on what you connect to it. A 1.2A (1200mA) power supply will provide you with ample power to run your Raspberry Pi for most applications, though you may want to get a 2.5A (2500mA) if you want to use all 4 USB ports without using an external powered USB hub. If you need to connect USB devices that will take the power requirements above 2.5A, then you must connect them to an externally-powered USB Hub.

It would be a good idea to get an USB car charger adapter (12v to 5V) with two outputs in case you need to power an USB Hub too. We recommend a power supply capable of providing a minimum of 3A.

Pay special attention to power supply as it is the main source of issues.

Buy a tested USB Power supply

http://www.sailoog.com/shop-category/openplotter

HDMI monitor

The Raspberry Pi has a HDMI port which you can plug directly into a monitor or TV with an HDMI cable.

The minimum size to use conveniently OpenPlotter is 7 inches (800x480px).

Actually this is not essential, you can use OpenPlotter from your laptop, tablet or phone by remote desktop. See chapter Headless.

Keyboard and mouse

Any standard USB keyboard and mouse will work with your Raspberry Pi. Wireless keyboard & mouse combos are a good option.

You will not need neither keyboard nor mouse if you are using OpenPlotter by remote desktop.

SD card

You need an SD card to work as the hard disk of your system and where you will install the operating system.

The Raspberry Pi 2 should work with any micro-SD-compatible card, although there are some guidelines that should be followed:

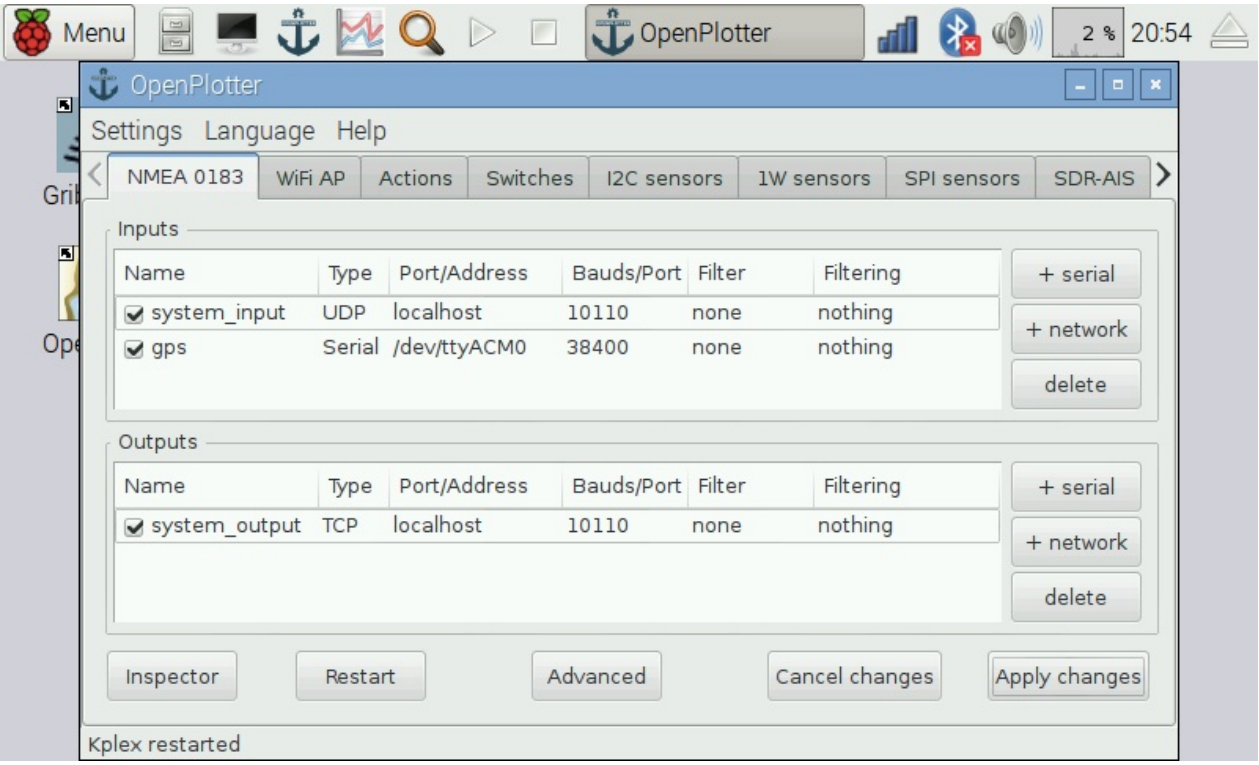
A minimum of 4GB is required but 8GB is recommended.

The card class determines the sustained write speed for the card; a class 4 card will be able to write at 4MB/s, whereas a class 10 should be able to attain 10 MB/s. However it should be noted that this does not mean a class 10 card will outperform a class 4 card for general usage, because often this write speed is achieved at the cost of read speed and increased seek times.

Buy an 8GB SD card with OpenPlotter RPI ready to run.

http://www.sailoog.com/shop-category/openplotter

Software



OpenPlotter RPI is a modified version of Raspbian, the official operating system for the Raspberry Pi. It contains all you need.

OpenPlotter RPI is open-source and free. Download and follow the instructions:

http://www.sailoog.com/en/blog-categories/openplotter-rpi

or buy our plug and play SD card with OpenPlotter RPI:

Buy an 8GB SD card with OpenPlotter RPI ready to run.

http://www.sailoog.com/shop-category/openplotter

Self powered USB Hub

If you are connecting devices which use more power than your Raspberry can provide, you will need a self powered USB Hub.

You can start connecting devices to the raspberry and switch to a self powered hub when you start to see strange behavior of the devices.

Hubs with **FE1.1S** chip work right, avoid hubs with MA8601 chip. Often the same model can contain any of them. And some of them do not contain chip at all! just a glob of black material to hide what is underneath ... nothing.

It would be a good idea to get a Hub with 5V input so that you could power it with the same source as your raspberry.

Buy a tested USB Hub

http://www.sailoog.com/shop-category/openplotter

USB WiFi dongle

You will need an USB WiFi dongle if you want to connect either OpenPlotter to internet or your mobile devices on board to OpenPlotter.

A good WiFi adapter will probably need more power than the Raspberry Pi USB port can provide, especially if there is a large distance from the WiFi adapter to the WiFi Access Point, or it is transferring large amounts of data. Therefore, you may need to plug the WiFi adapter into a powered USB hub.

To share data with on board devices by WiFi you have to set OpenPlotter as an access point and connect devices to it. However not all WiFi dongles can function as an access point, only devices with the **RT5370**, **RTL8188CU/CUS** or **RTL8192CU/CUS** chipset will work (OpenPlotter RPI v0.6.0 supports **RTL8192EU** too). We recommend **RTL8192CU/CUS**.

Buy a tested WiFi dongle

http://www.sailoog.com/shop-category/openplotter

USB GPS dongle

If you don't have any GPS on board or you want an extra positioning device, this is the cheapest and most effective way.

Connecting an USB GPS dongle to OpenPlotter will provide accurate position, date/time and speed/course over ground.

Buy a tested GPS/GLONASS USB dongle

http://www.sailoog.com/shop-category/openplotter

NMEA 0183 to USB converter

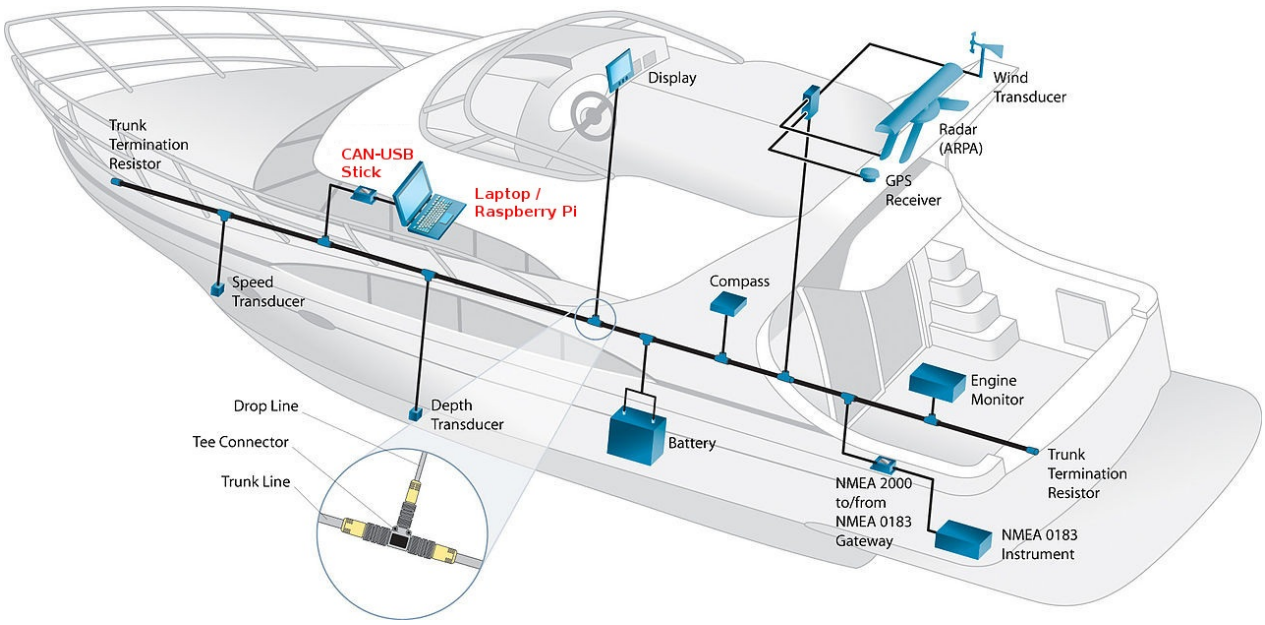
If you have electronics with NMEA 0183 outputs on board (depth, wind, heading...), you will need an USB converter to connect it to OpenPlotter. Additionally, if this converter is bi-directional, you will be able to talk to electronics with NMEA 0183 inputs like the autopilot.

The NMEA 0183 hardware standard uses **RS422** connectors but you may find some devices with **RS232** as well. Find out about what type of connection you need.

Buy a tested USB-RS422 bi-directional converter

http://www.sailoog.com/shop-category/openplotter

CAN-USB Stick



You can buy a CAN-USB Stick converter here

http://www.sailoog.com/shop-category/openplotter

Warning / Disclaimer

The CAN-USB Stick is a research project on data communication in CAN bus and N2K networks on boats.

The software is still under development and has not been fully tested. Malfunctions of the CAN-USB Stick and of any connected device might be possible at any time. Manipulating your N2K network could cause damage to connected devices.

Do not rely on data from this device and do not use as primary source for navigation. Liability cannot be accepted for any damages, personal injuries or malfunctions caused by this device.

The CAN-USB Stick is not certified by NMEA®.

The CAN-USB Stick is not electrically isolated.

It is not allowed to use the Actisense® NMEA Reader software for the CAN-USB Stick.

Overview

The CAN-USB Stick is based on a stm32F103 micro-controller (MCU) connected to a CAN transceiver and is able to read/write on the CAN bus in a boat network.

The stick works with a CH340 USB to serial converter. Windows 10 does recognize the serial port (no driver is needed). Linux does also support CH340.

N2K network is described in Wikipedia https://en.wikipedia.org/wiki/NMEA_2000

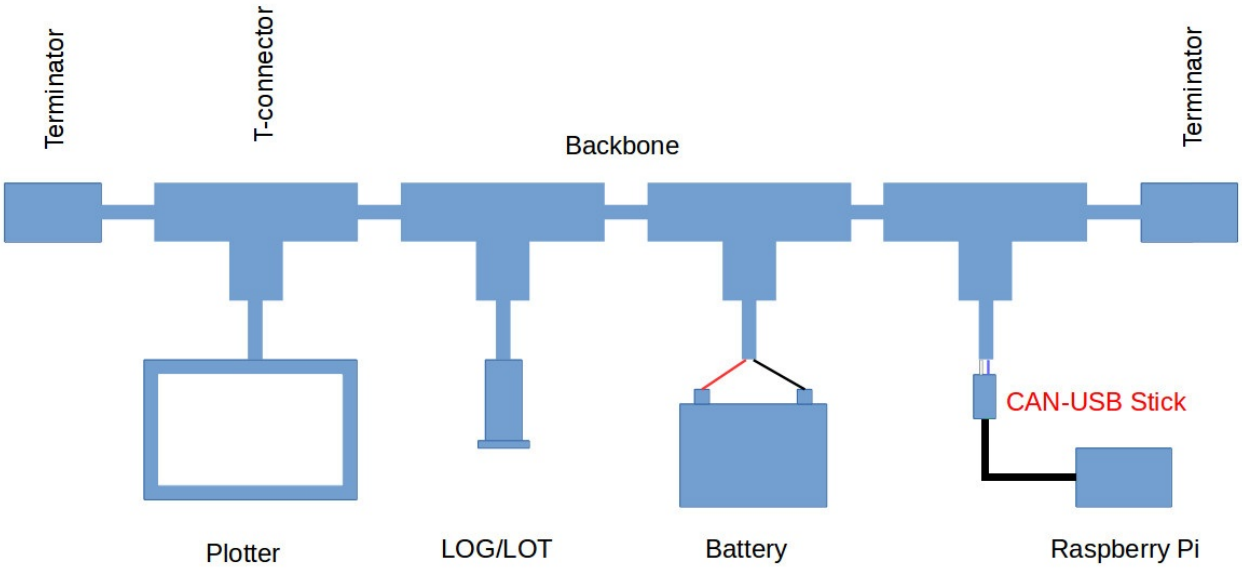
The program of the MCU has been re-engineered to work together with CANBOAT project (https://github.com/canboat), which is used by Signal K project (http://signalk.org).
Both packets are used in OpenPlotter project.

The stick does also work with OpenSkipper project (http://openskipper.org).

Not tested:

- MacENC (http://macenc.com)
- PolarView NS (http://www.polarnavy.com)

N2K networks



Example of a small N2K Network

The backbone (or trunk) starts with a 120Ω terminator and ends with a 120Ω terminator. The two resistors are working in parallel, so the resistance is 120Ω/2=60Ω. If there is a broken connection in the backbone you can measure only 120Ω or nothing but not 60Ω. That is a very easy way to check the bus.

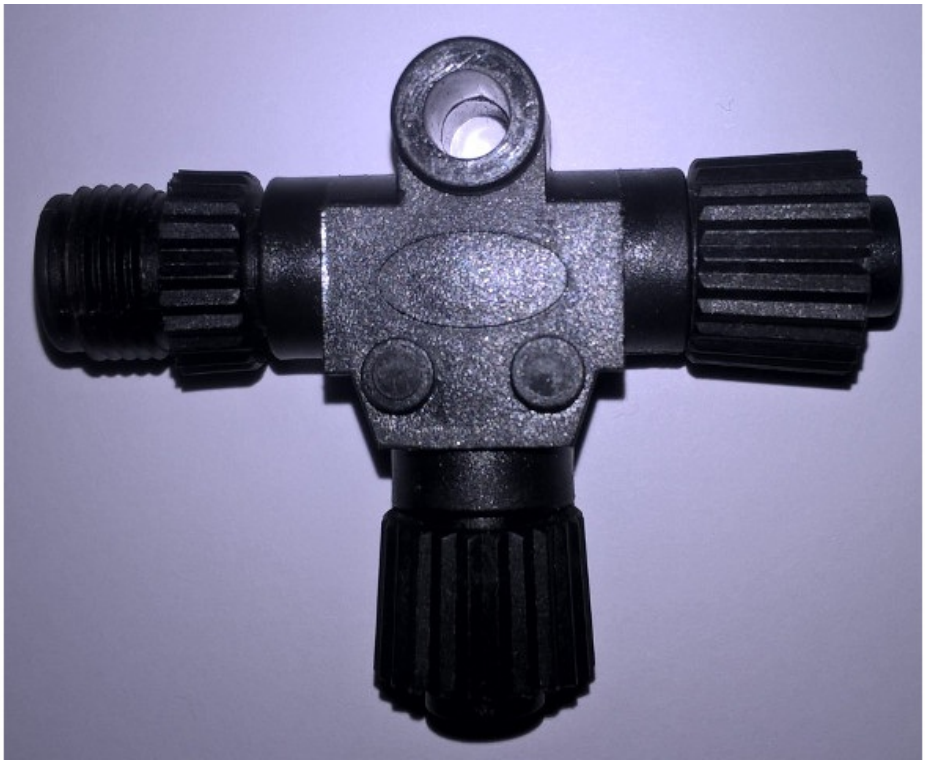
M12 male 120Ω terminator

The drop line to the devices should not be longer than 6 m. The backbone can have 100m in length.

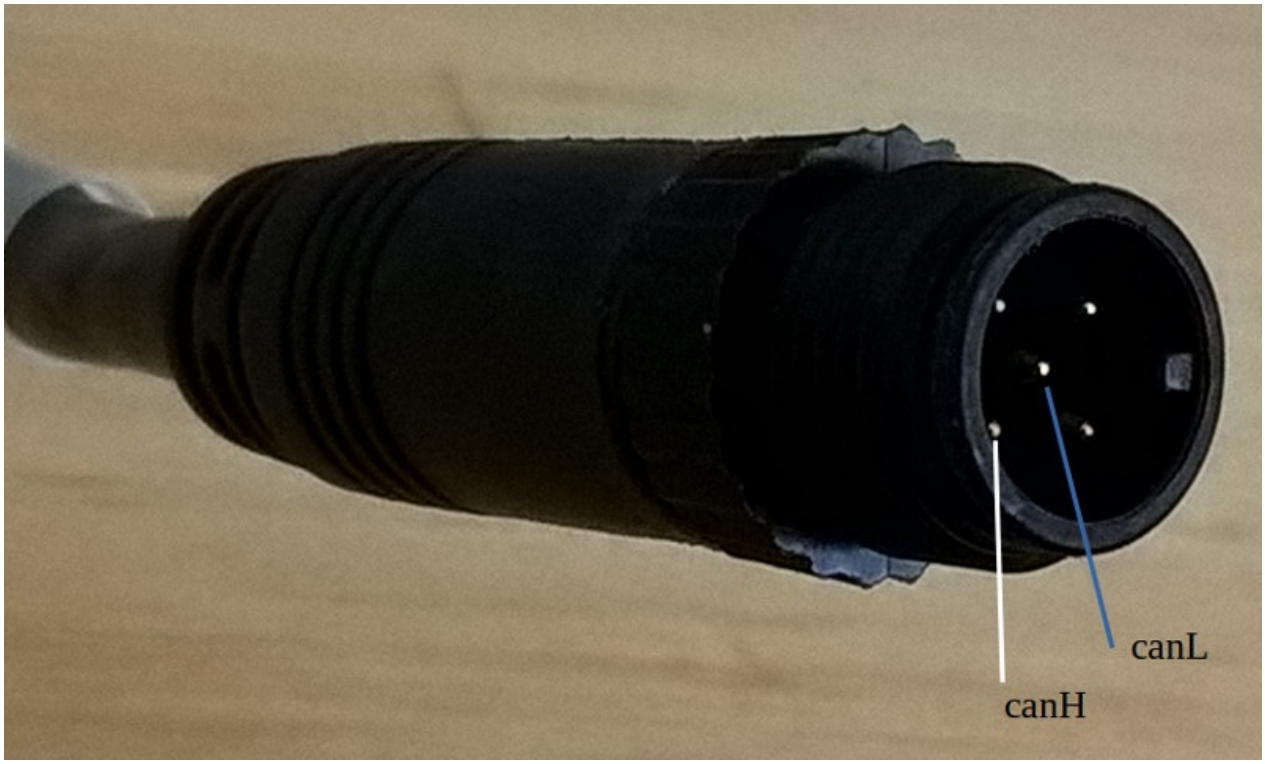
The CAN-USB Stick is not electrically isolated, so connecting the Raspberry Pi/computer power supply to the same CAN bus power supply is recommended.

Connection

To connect the CAN-USB Stick to the network you need a free T-connector on your backbone and a drop line. The drop line should have a M12 5 pin male connector in one side and 5 wires (but we only need 2) in the other side. The HIRSCHMANN ELST 5012 PG7 connector has a screw terminal.

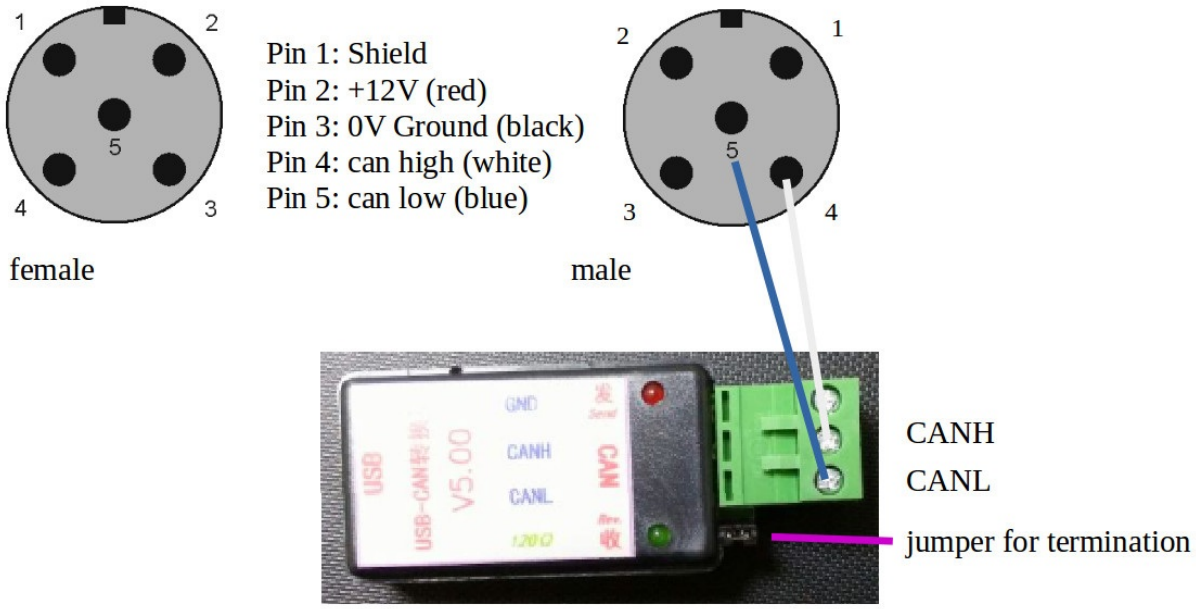


T-connector



Drop line M12 5 pins male connector side

Drop line wires side



- Pull out the green screw terminal of the stick.
- Connect the drop line blue wire from pin 5 (pin in the middle) to the green terminal on CANL.
- Connect the drop line white wire from pin 4 to the green terminal on CANH.
- Turn off the main power switch to be sure that there is no power on the network.
- Connect the drop line to the free T-connector on your backbone.
- Use a multimeter and measure the resistance between CANH and CANL (on the screws). The resistance should be around 60 Ohm.
- Pull of the jumper for termination from the stick (the network should be already terminated).
- Connect the green screw terminal to the CAN-USB Stick.
- Check again the 60 Ohm between CANH and CANL.
- On the drop line there are three cables left. They have to be isolated.
- Turn on the main power.
- Switch on instrumentation.

To check the N2K traffic on the bus, see the chapter Signal K.
On other Linux distribution, use the canboat command: `actisense-serial -r /dev/ttyUSBx (x: use the right device number, typical 0). On Windows use OpenSkipper.`

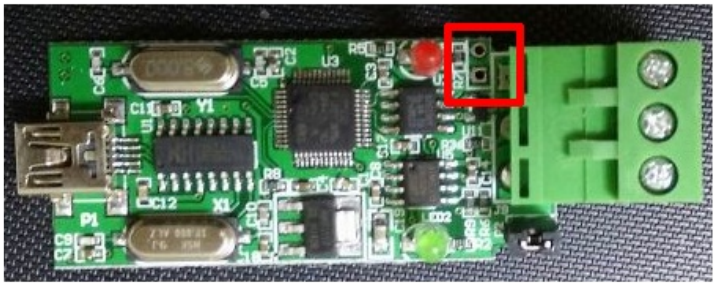
	Red LED	Green LED
Bootsetup (8")	off	off
Connected only to computer	flashing every second	on
Connected to bus	flashing every second	flashing on traffic

Software

The software is still under development and has been built under Eclipse mars1 with the GNU ARM Eclipse Windows Build Tools. The software is licensed under apache 2.

We will publish regular updates. The built .hex file can be flashed into the stick with the “Flash Loader Demonstrator” from ST on Windows (http://www.st.com/web/en/catalog/tools/PF257525).

The board has an unused place for a jumper. Connecting the two holes of the not molded jumper will put the stick into boot loading mode.



Support

http://forum.openmarine.net/

USB DVB-T dongle

DVB-T dongles based on the Realtek **RTL2832U** chip can be used as a cheap one channel AIS receptors.

A DVB-T dongle will need more power than the Raspberry Pi USB port can provide. You need to plug the dongle into a self powered USB hub.

Antenna

The most important factor for good reception is the antenna. Any VHF antenna will work right. Some proficient homemade antennas:

http://sdrformariners.blogspot.com.es/p/blog-page.html

http://nmearouter.com/docs/ais/aerial.html

https://www.youtube.com/watch?v=SdEglNHyHB4

Buy a tested USB DVB-T dongle

http://www.sailoog.com/shop-category/openplotter

IMU sensor

If you don't have a electronic compass on board you will need an IMU.

An Inertial Measurement Unit, or IMU, measures and reports on velocity, orientation and gravitational forces, using a combination of an accelerometer, gyroscope, and a magnetometer.

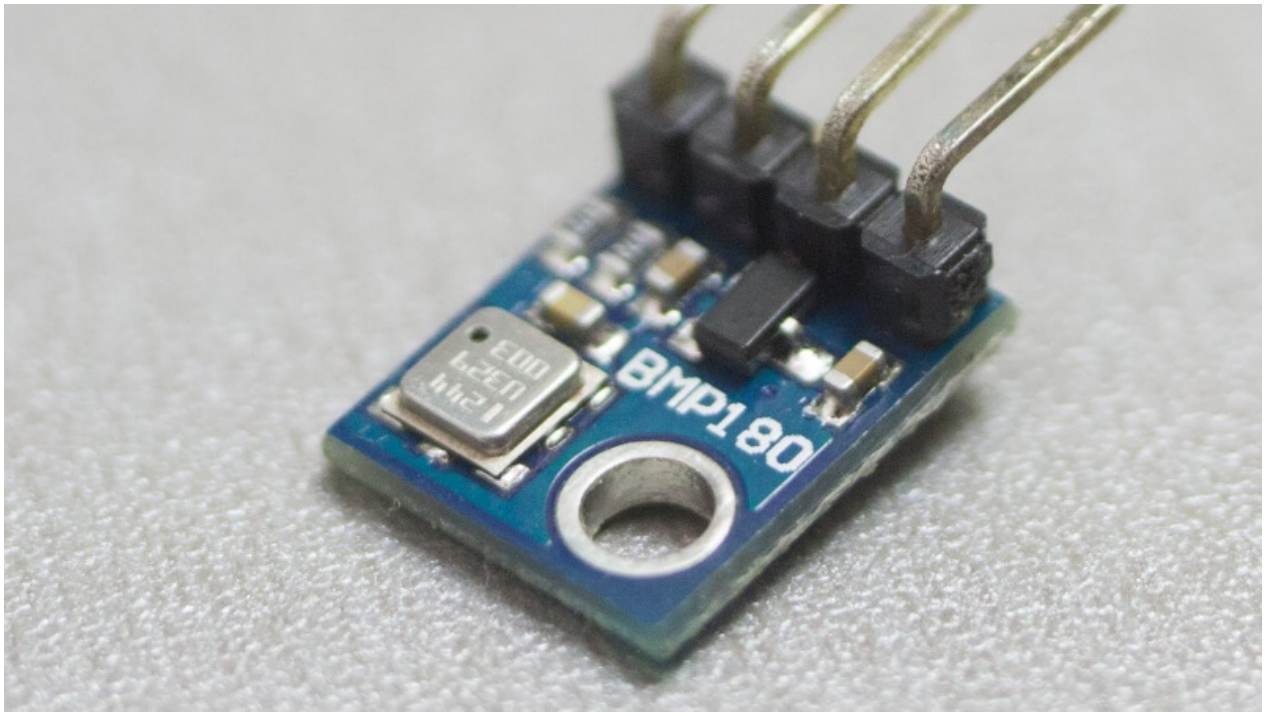
Connecting an IMU to OpenPlotter will provide magnetic heading which is needed to calculate true heading and true wind. You will have heel angle data as well.

IMU sensors have to be connected by I2C interface.

Supported IMU sensors

- InvenSense MPU-9150 single chip IMU.
- InvenSense MPU-6050 plus HMC5883 magnetometer on MPU-6050's aux bus (handled by the MPU-9150 driver).
- InvenSense MPU-6050 gyros + acclerometers. Treated as MPU-9150 without magnetometers.
- InvenSense MPU-9250 single chip IMU (I2C and SPI).
- STM LSM9DS0 single chip IMU.
- STM LSM9DS1 single chip IMU.
- L3GD20H + LSM303D (optionally with the LPS25H) as used on the Pololu AltIMU-10 v4.
- L3GD20 + LSM303DLHC as used on the Adafruit 9-dof (older version with GD20 gyro) IMU.
- L3GD20H + LSM303DLHC (optionally with BMP180) as used on the new Adafruit 10-dof IMU.
- Bosch BMX055 (although magnetometer support is experimental currently).
- Bosch BNO055 IMU with onchip fusion. Note: will not work reliably with RaspberryPi/Pi2 due to clock-stretching issues.

Pressure/Temperature sensor



Connecting a pressure sensor to OpenPlotter will provide air pressure data to build graphs and monitor weather.

Often, pressure and temperature sensors are on the same board.

If you have another board like an humidity sensor which has a temperature sensor too, you will be able to select which one you prefer.

Pressure/Temperature sensors have to be connected by I2C interface.

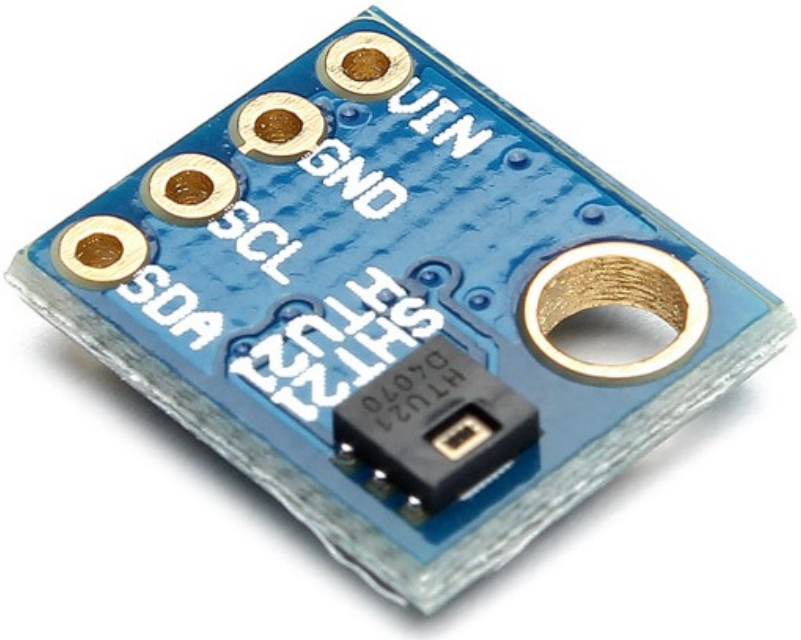
Supported pressure/temperature sensors

- BMP180
- LPS25H
- MS5611
- MS5637

Buy a tested BMP180 sensor

http://www.sailoog.com/shop-category/openplotter

Humidity/Temperature sensor



Connecting a humidity sensor to OpenPlotter will provide air relative humidity data to build graphs and monitor weather.

Often, humidity and temperature sensors are on the same board.

If you have another board like a pressure sensor which has a temperature sensor too, you will be able to select which one you prefer.

Humidity/Temperature sensors have to be connected by I2C interface.

Supported humidity/temperature sensors

- HTS221
- HTU21D

Buy a tested HTU21D sensor

http://www.sailoog.com/shop-category/openplotter

One wire temperature sensor

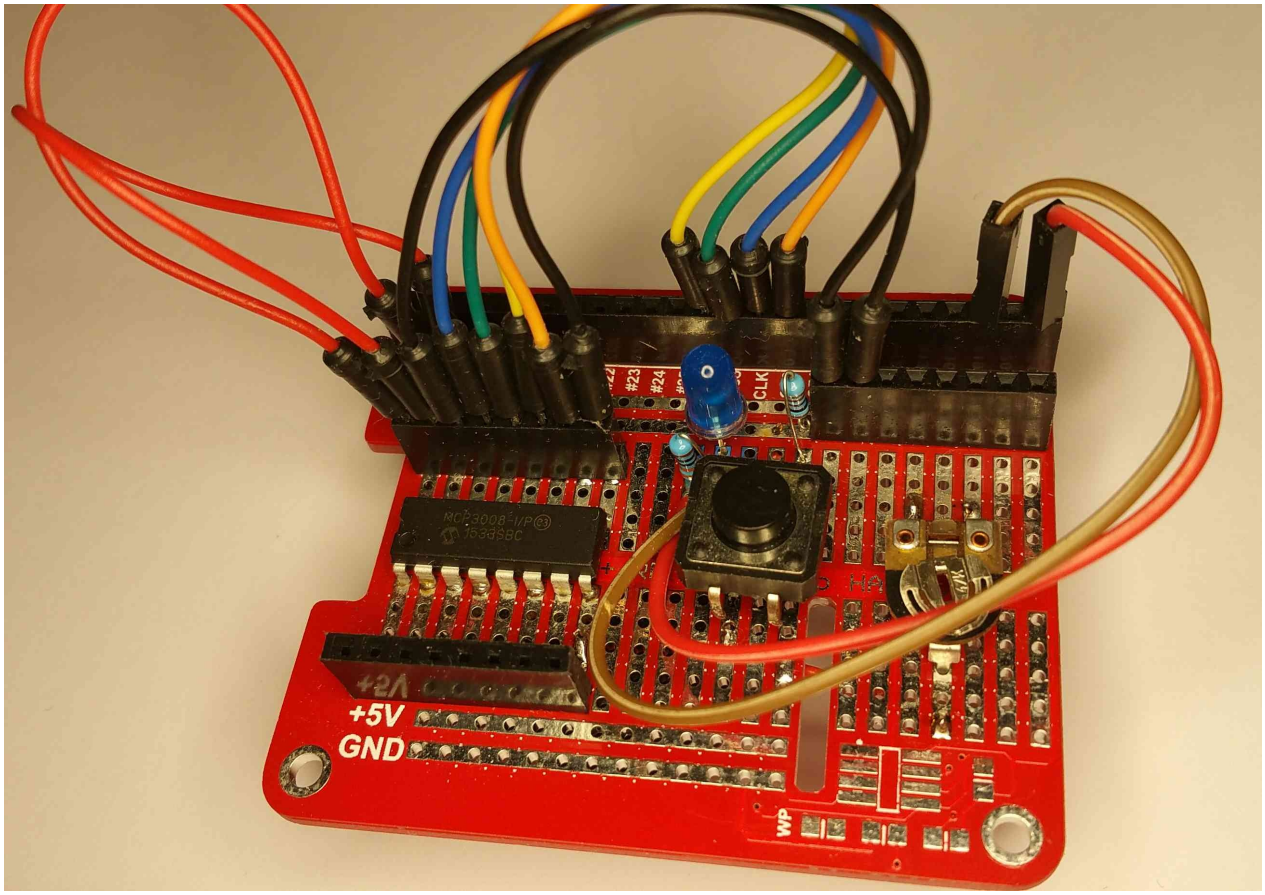
You can connect one wire **DS18B20** sensors to OpenPlotter. These sensor are waterproof and can withstand high temperatures. Connecting multiple DS18B20 in parallel to the same pins, you will be able to get temperature data from coolant engine, exhaust, engine room, fridge, sea ...

Buy a tested DS18B20 sensor

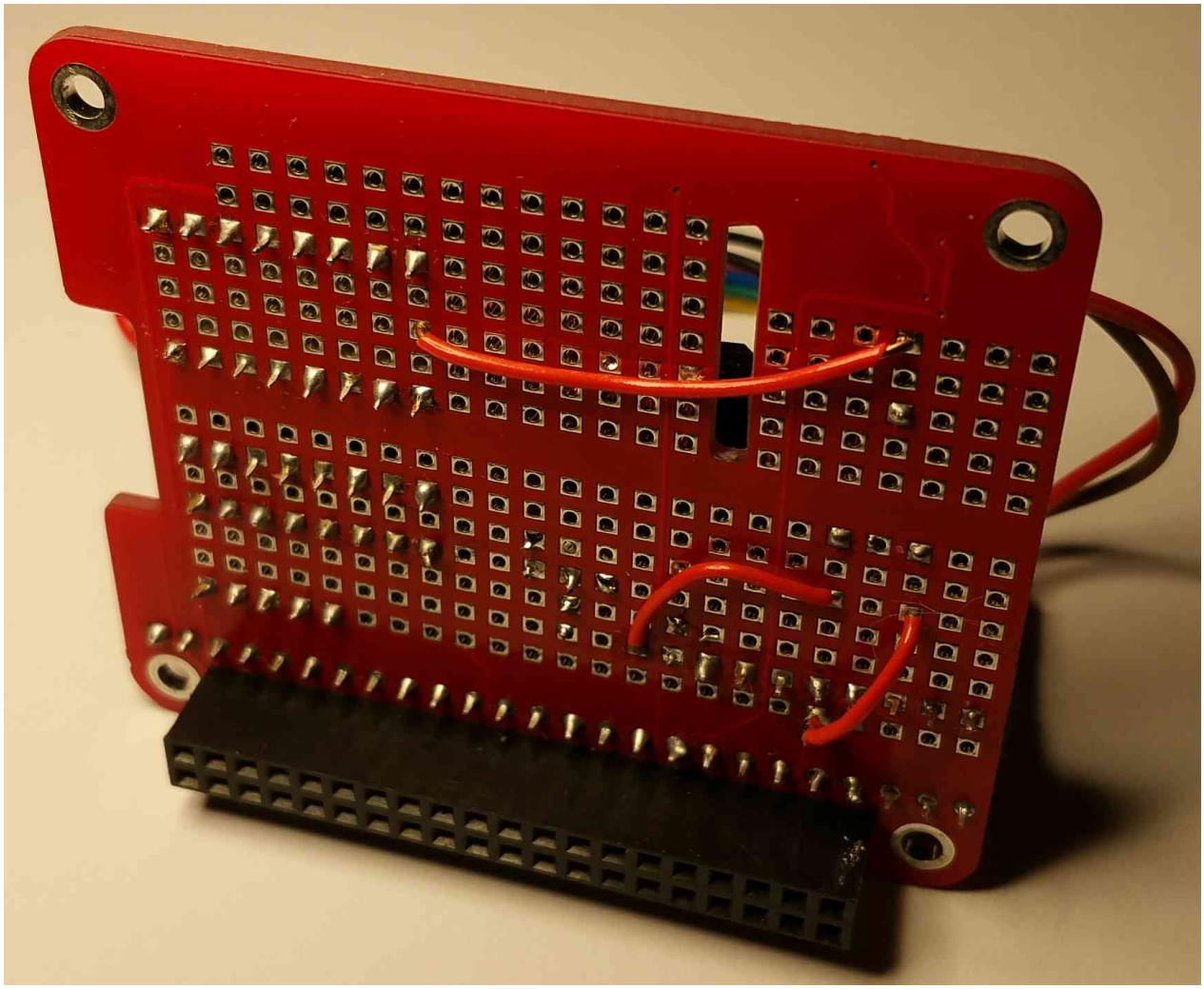
http://www.sailoog.com/shop-category/openplotter

Analog Digital Converter

MCP3008

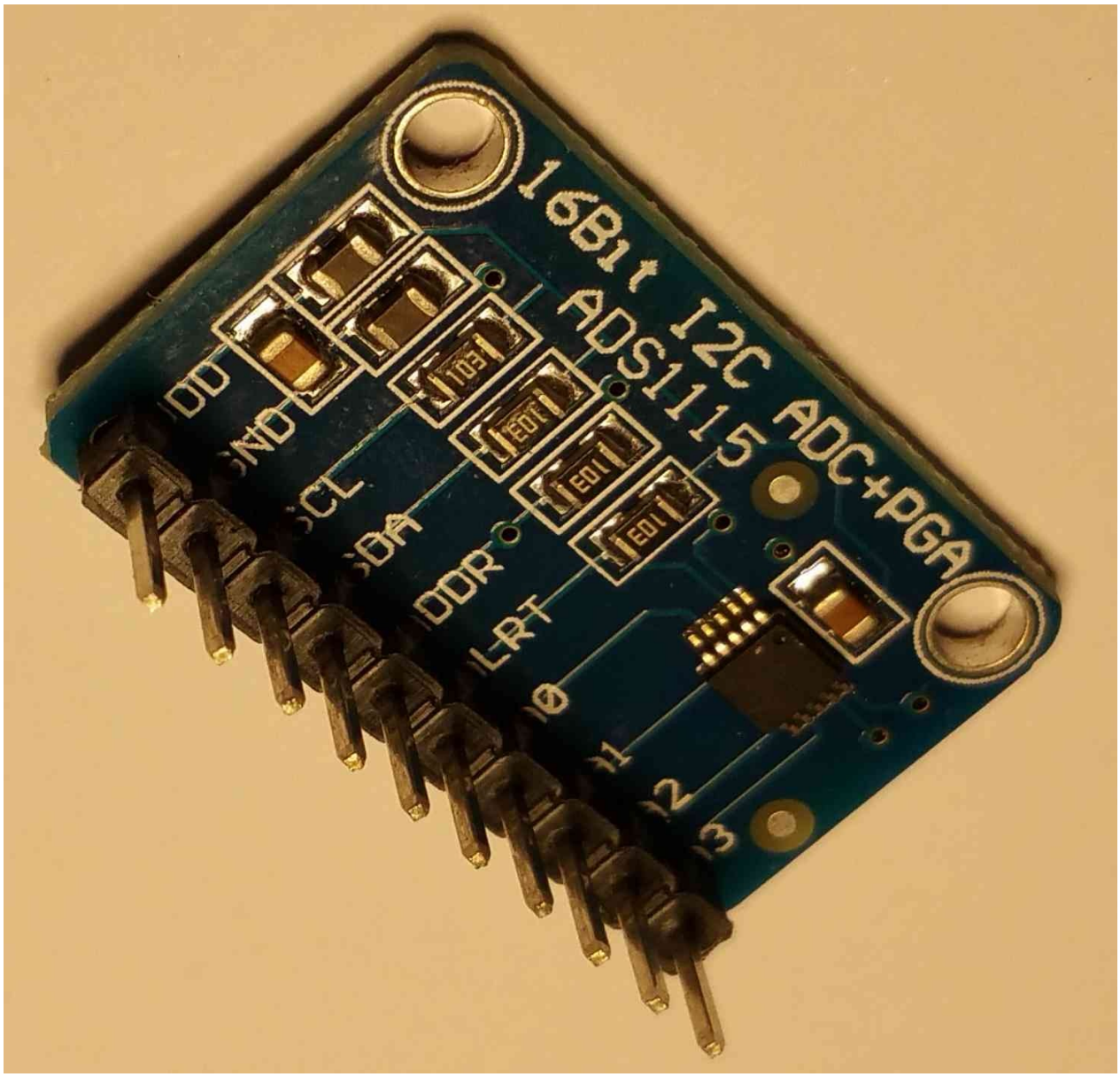


DIY development board front



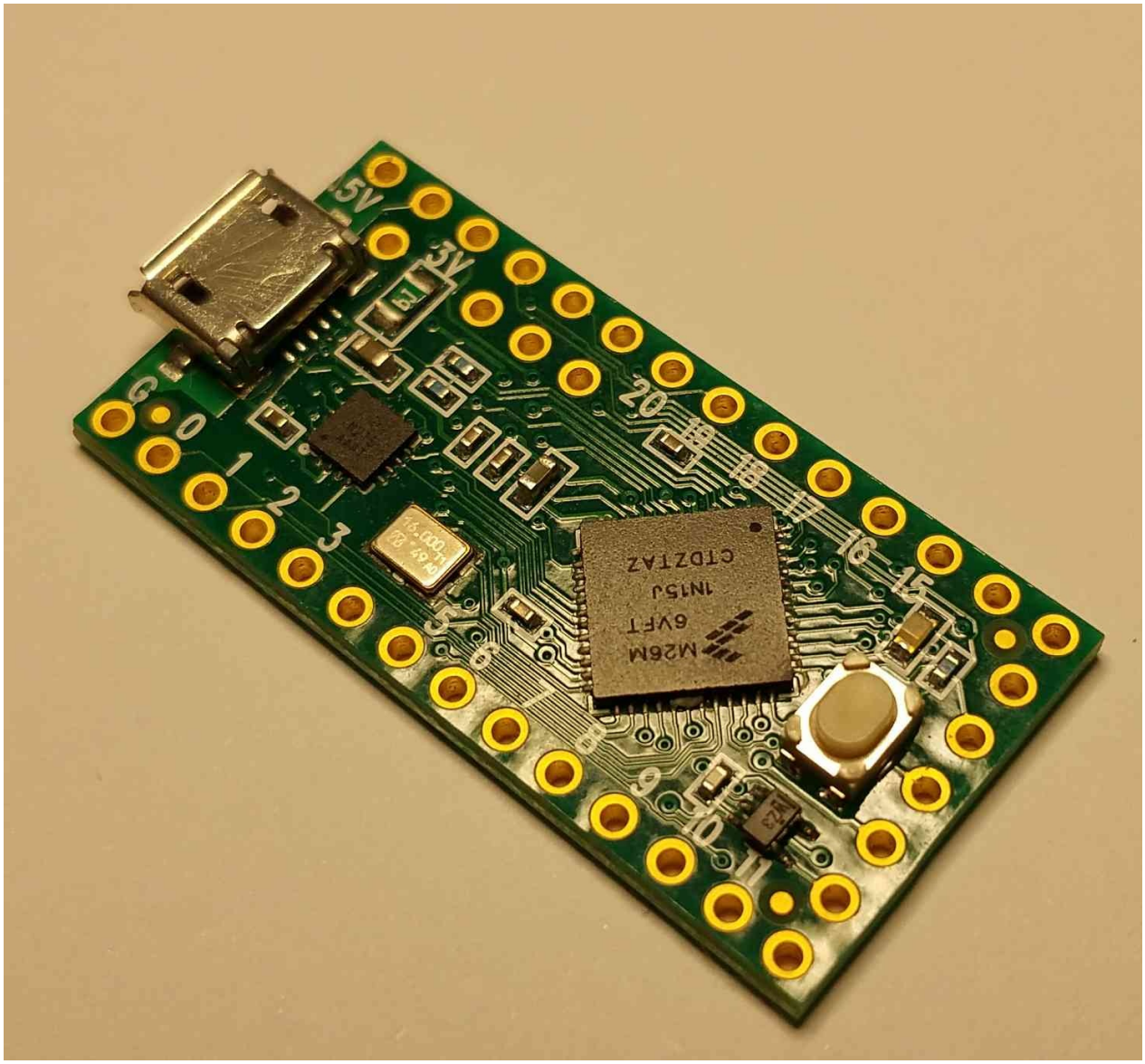
DIY development board back

ads1115



has to be connected to I2C bus

use the adc of the MCU with Firmata protocol



Any Arduino IDE programable board can be used (Picture: Teensy LC)

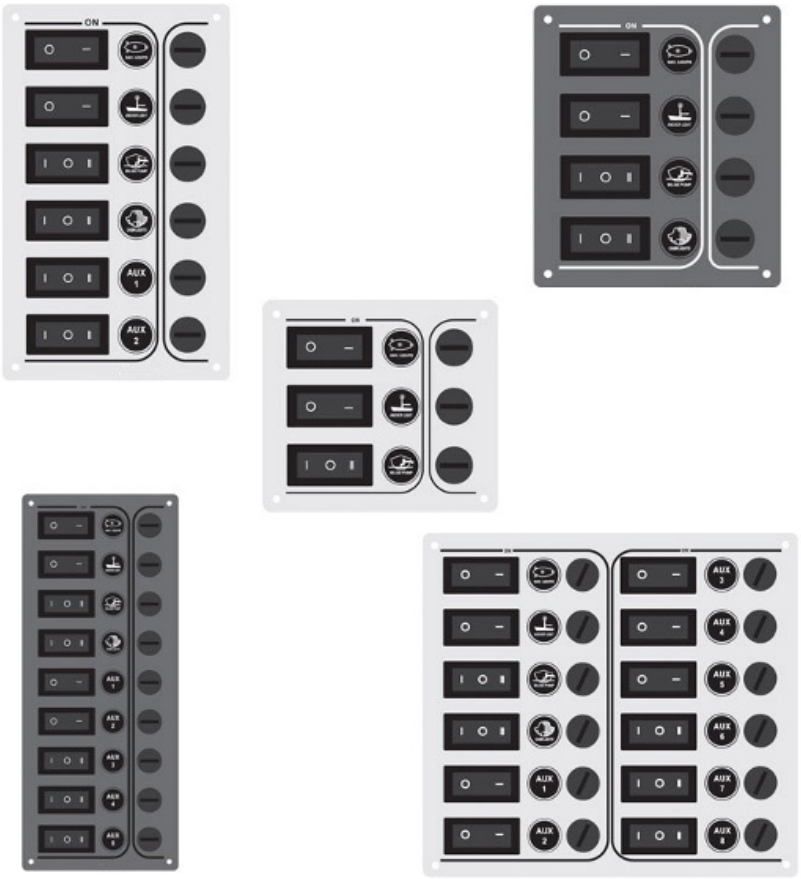
PIR motion sensor

Connecting this sensor you can detect if a human has moved in or out of the sensor range. They are small, inexpensive, low-power and easy to use. It is basically made of a pyroelectric sensor which can detect levels of infrared radiation. Everything emits some low level radiation and the hotter something is, the more radiation is emitted. If you connect OpenPlotter to internet you will be aware of what is happening in your boat when you are not there.

Buy a tested PIR motion sensor

http://www.sailoog.com/shop-category/openplotter

Common switch



Almost all of the OpenPlotter features have a defined action. You can connect any type of external switch (momentary, toggle...) and assign an action to each state (on/off). So you can shutdown/reset the system, run a custom Linux command, play/stop services, trigger external devices, play/stop an audio ...

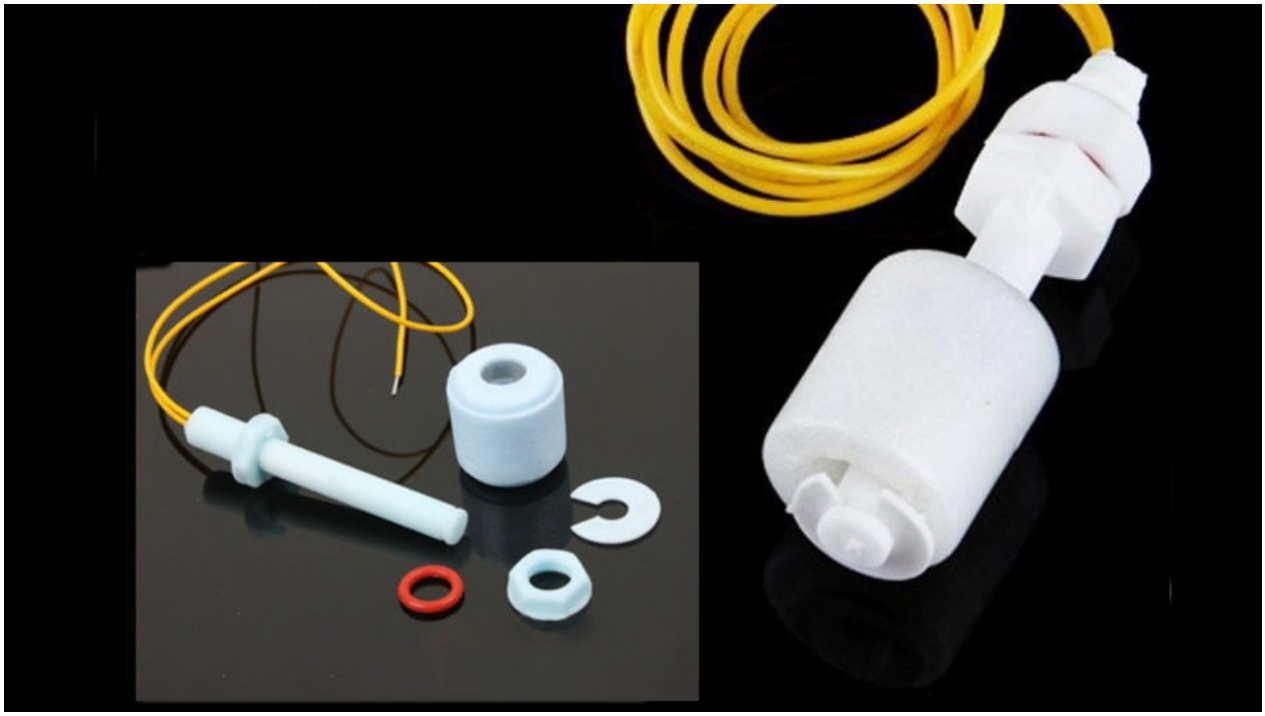
Door switch

Installing this little magnetic switches on doors and windows you will be able to detect any unauthorized opening. If you connect OpenPlotter to internet you will be aware of what is happening in your boat when you are not there.

Buy a tested Door switch

http://www.sailoog.com/shop-category/openplotter

Float switch



Installing this little magnetic switches on tanks and bilges you will be able to detect when they are full or empty. OpenPlotter may actuate a pump, an indicator, an alarm or warn you by Twitter or email.

Buy a tested Float switch

http://www.sailoog.com/shop-category/openplotter

Relay

This chapter is under construction

LED

This chapter is under construction

Buzzer

This chapter is under construction

Getting started

First of all you have to put together all the required parts. If you have trouble with some aspect, try to find help on the Raspberry Pi official page.

Second you have to run the software on your ARM computer and here you have two options, either buy our plug and play SD card or download and install the software on an SD card.

Buy an 8 GB SD card with OpenPlotter RPI ready to run.

http:\/\/www.sailoog.com\/shop-category\/openplotter

Installing OpenPlotter RPI on an SD card

Download the last noobs version of **OpenPlotter RPI** from

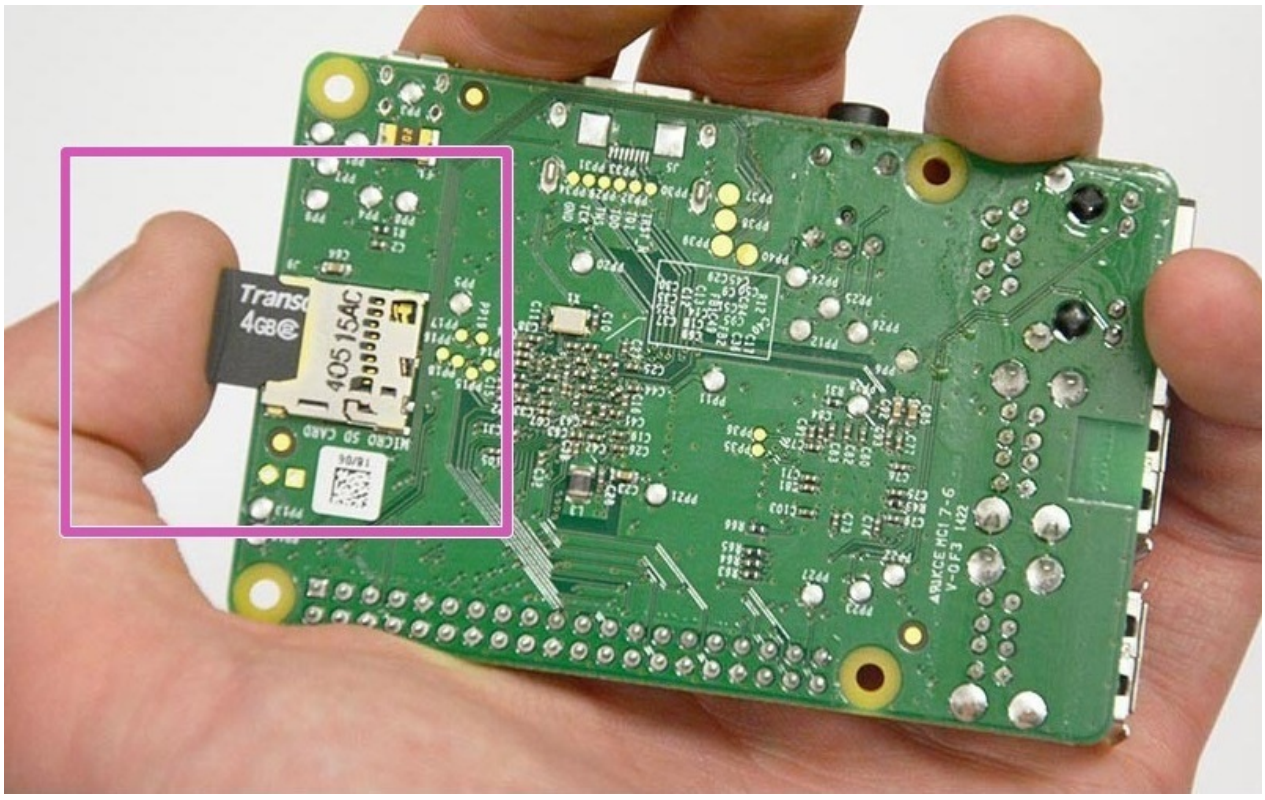
http:\/\/sailoog.com\/blog-categories\/openplotter-rpi

It is a compressed file ca. 1GB so it will take a while.

Once the download is completed we have to unzip it and copy all files from the noobs folder onto a micro SD FAT32 formated card minimum 8 GB.

If you bought our SD card you just from here.

Now we will insert the created or bought SD card with OpenPlotter RPI into our Raspberry Pi.



First boot

If you want to build a headless system see the next chapter Headless before reading further.

Connect power to the RPI.

Now we wait for the noobs menu.

In the noobs menu we select openplotter and click install. It takes several minutes to format and install the system.

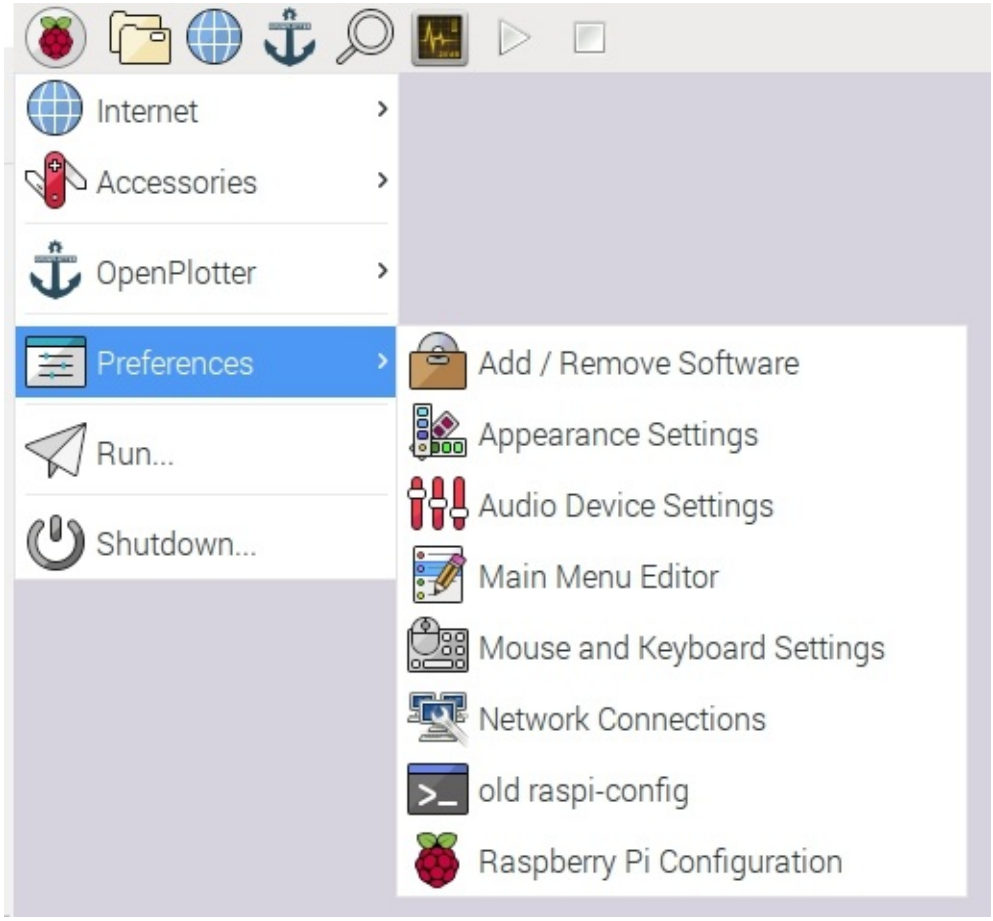
When noobs has finished openplotter starts.

AutoSetup pops up (We can skip or close it. It will appear on every boot until we uncheck setup autostart on every boot).

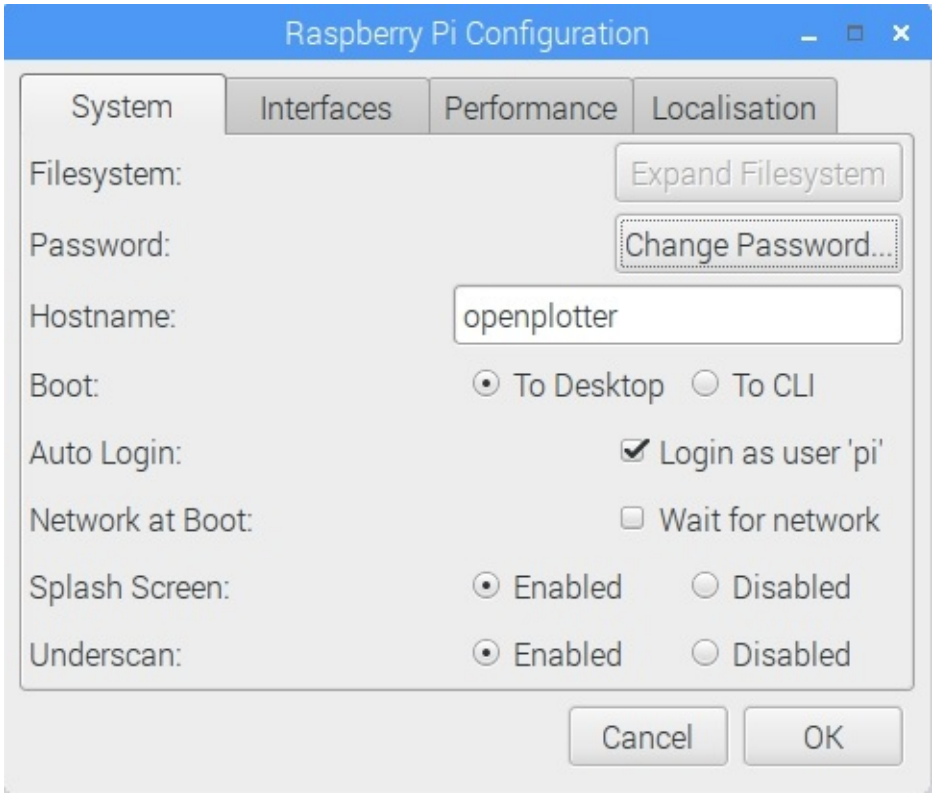
The nativ monitor resolution for 800x480 monitors will be auto detected. The right settings for it will work on the next boot! If we have such a monitor. We do a restart.

Personal settings

Go to *Menu* > *Preferences* and select *Raspberry Pi Configuration*.



A window will open and it's a good idea to change the Password to make OpenPlotter more secure. Click on *Change Password* (default password: raspberry).



Setting language and country

You need to set your system localisation, click on the *Localisation* tab and then on *Set Locale*, *Set Timezone, Set Keyboard, Set WiFi Country* buttons.

To configure openplotter we start with setting up the usb adapter Auto Setup USB ports.

Security

Because of security reasons **ssh is disabled by default** and can be activated under Interfaces.

Then we can use:

on windows system PuTTY and WinSCP

on linux system putty and nautilus

to exchange files and get a remote terminal.

Keep it disabled if you don't need it!

Headless

You can use the RPI in headless mode after the noobs installation is done.

There are two ways to work headless.

1. Over wifi (You have to setup a wifi access point (hotspot) WiFi access point.
2. You can connect to the RPI by an Ethernet cable and use the bonjour protocol or the bridged hotspot.

Once we have created the SD card with noobs, we have to insert it into any computer with any OS (Linux, MAC, Windows). The mounted device should be called *boot* and there should be a file called *config.txt*. Open this file in a text editor like notepad on windows, but not anything like MS Word which can save it as something which is not just plain text.

The top lines should look like this:

```
[OPENPLOTTER]

#HEADLESS

# uncomment to enable WiFi access point. Default=0
#wifi_enable=0
# uncomment to set WiFi access point. wlan0, wlan1, wlan2 ... default=wlan0
#device=wlan0
# uncomment to set WiFi access point. SSID must have a maximum of 32 characters, default=OpenPlotter
#ssid=OpenPlotter
# uncomment to set WiFi access point. Password must have a minimum of 8 characters, default=12345678
#pass=12345678
# uncomment to set WiFi access point. b=IEEE 802.11b, g=IEEE 802.11g, defaul=g
#hw_mode=g
# uncomment to set WiFi access point. 1 to 11, default=6
#channel=6
# uncomment to set WiFi access point. 1=WPA, 2=WPA2, 3=both, default=2
#wpa=2
# uncomment to select device connected to Internet and share connection. wlan0, wlan1, wlan2 ... default=0
#share=0
# uncomment to enable bridge to eth0. Default=0
#bridge=0
# uncomment to set the access point IP. Default=10.10.10.1
#ip=0

# uncomment to force screen resolution (only VNC remote desktop)
#framebuffer_width=800
#framebuffer_height=600

...
```

If you are going to connect to OpenPlotter by VNC remote desktop, you have to remove the # character from words ***framebuffer_width*** and ***framebuffer_height*** and set the screen size (800x600 by default).

If you are going to connect to OpenPlotter by RDP remote desktop, you do not have to remove the # characters from words *framebuffer_width* and *framebuffer_height*, you will set the screen size from RDP software. See more information in Remote desktop chapter.

To create the WiFi hotspot you have to remove the # character from words ***device***, ***ssid*** and ***pass***.

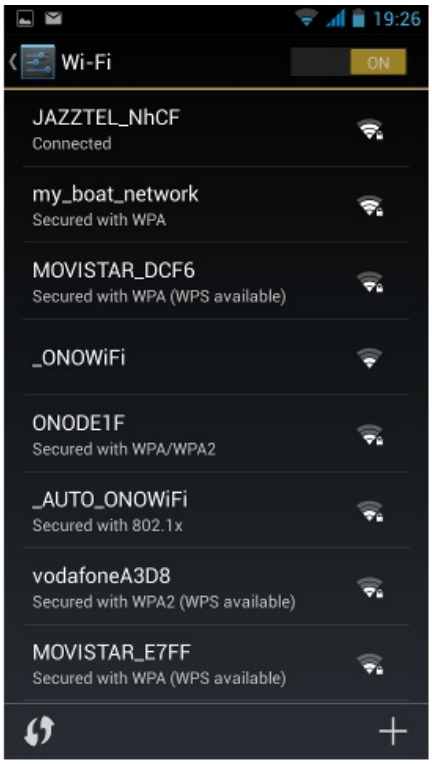
If only one WiFi dongle is connected, the *device* value should always be *wlan0* but if more than one is connected, the *device* value could be *wlan0*, *wlan1* ...

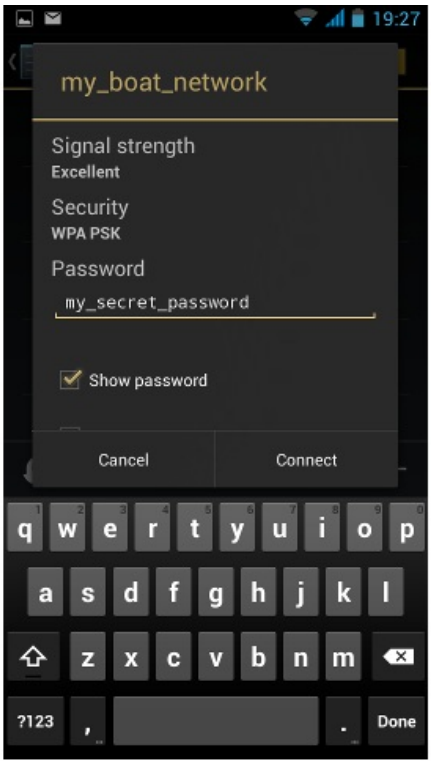
ssid will be the name of your WiFi network. Use any character but a maximun of 32.

pass will be the password of your WiFi network. Use any character but a minimum of 8.

Save and eject the SD card, then insert it into the Raspberry Pi. Make sure the WiFi dongle is also inserted and power up the Pi.

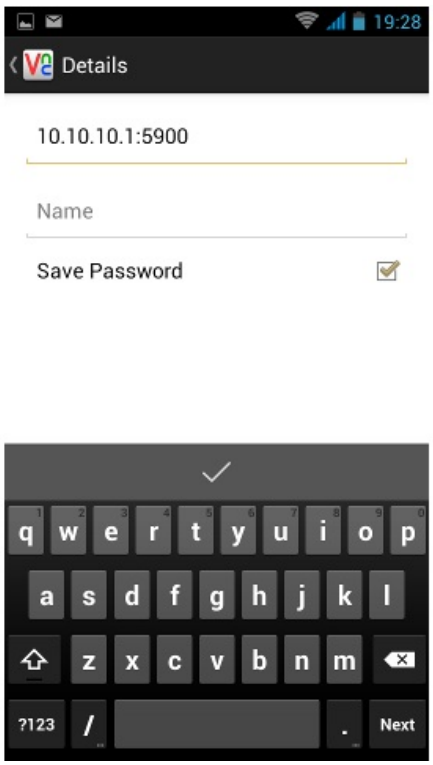
After it has finished booting up, there should be a new WiFi network called *OpenPlotter* available. Log onto this network using the password *my_secret_password*.

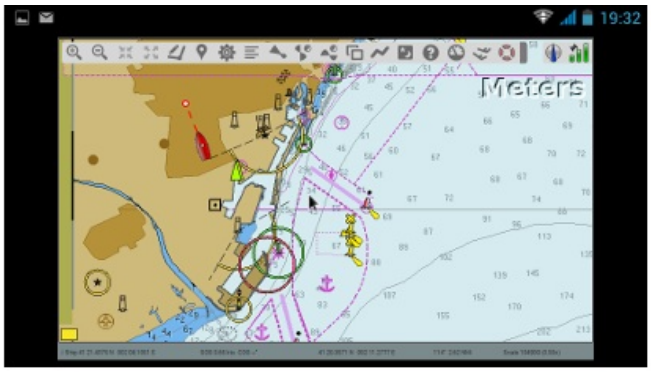


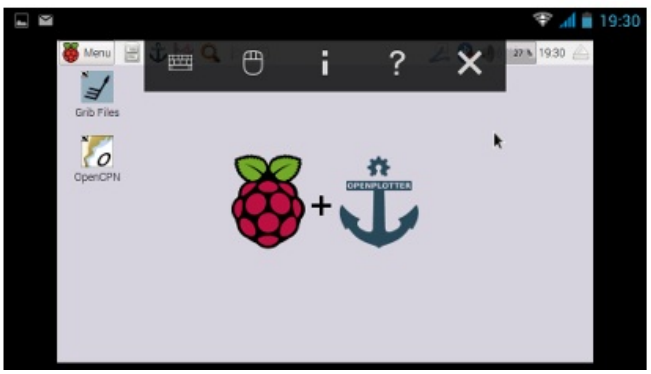


Finally, open your favourite VNC remote desktop client on your laptop, tablet or smartphone and make a new connection with the address/port combination: **10.10.10.1:5900**, or just the address **10.10.10.1** if you are using a RDP remote desktop client.

Connect and enjoy!







See more information about tested remote desktop software in Remote desktop chapter.

If this is your first boot, go to Getting Started chapter and follow the steps.

Remote desktop

To connect to the raspberry pi openplotter can setup an AP (access point) or and can be connected by an ethernet cable.

It can be easily connected by bonjour protocol with "openplotter.local" as address or use the IP address.

The IP address of openplotter as WiFi access point is **10.10.10.1** as default.

In \/home\/pi\/.config\/openplotter\/openplotter.conf you can change the address manually.

If you connect the raspberry pi to a router by ethernet cable (and bridge mode isn't checked) the raspberry pi will get the IP address from the routers dhcp server. So connect to the router and look what IP address it sets for openplotter.

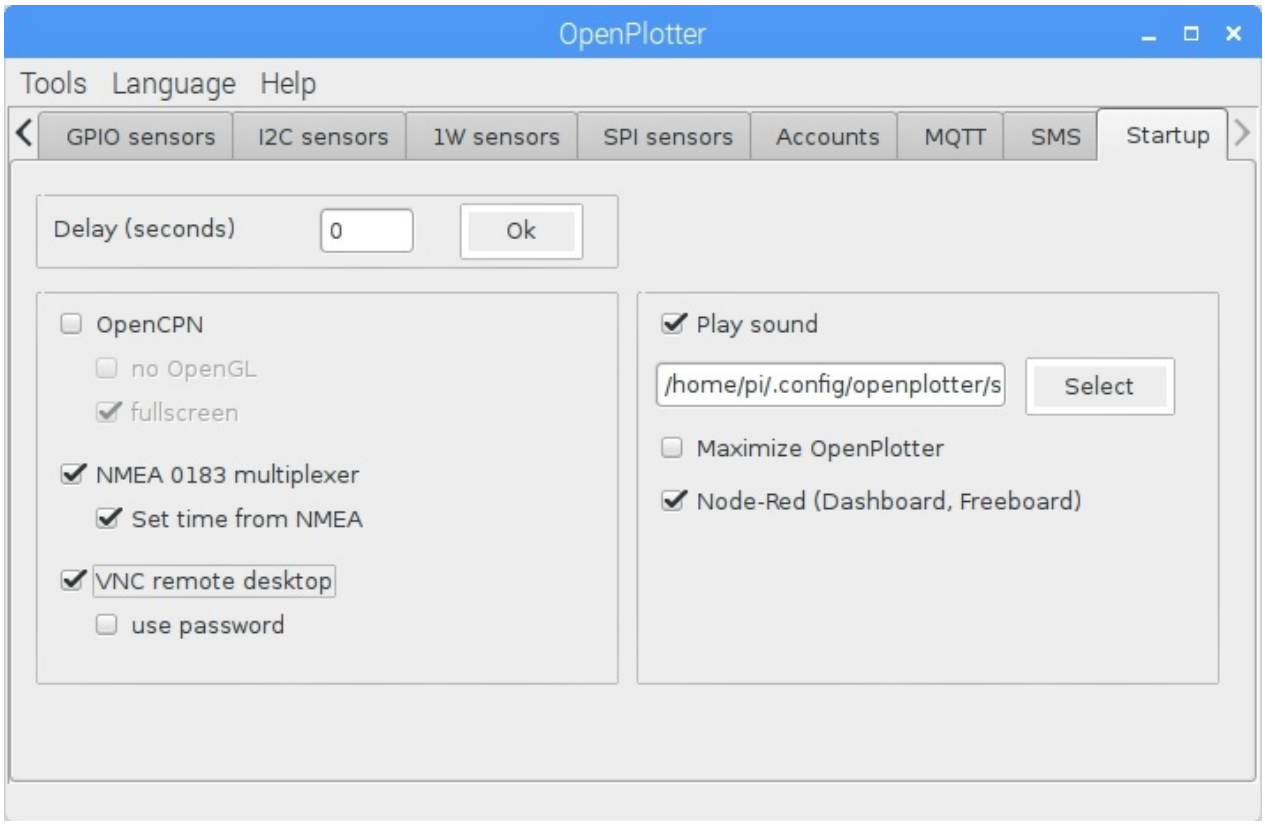
You have to install a remote desktop client on your remote device. There are two types, VNC and RDP, and some applications can use both of them.

VNC replicates the current desktop in OpenPlotter.

RDP can replicate the current desktop or create a new one with different resolution if you want. RDP is faster than VNC.

VNC

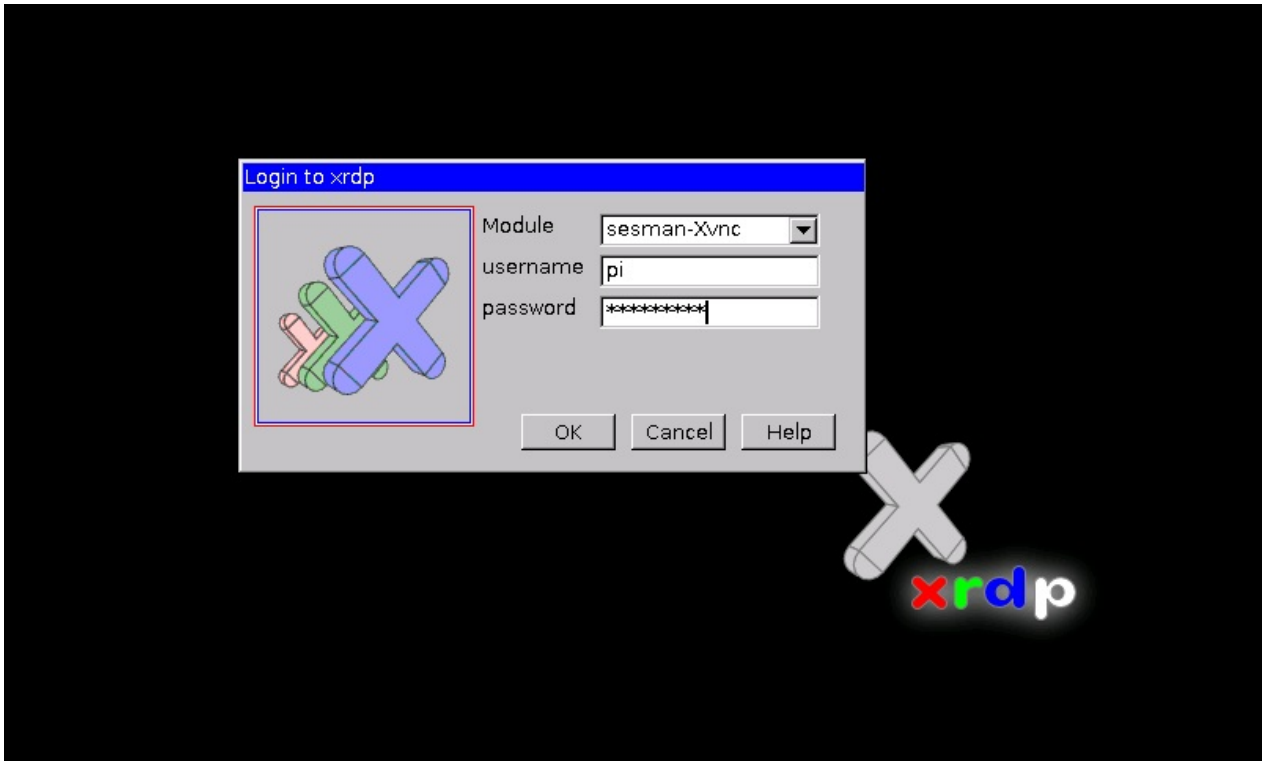
Be sure the checkbox *VNC remote desktop* is enabled in the *Startup* tab.



To connect by VNC you have to provide the IP of OpenPlotter and the port 5900.

RDP

To connect by RDP you have to provide just the IP or openplotter.local. Then you will be prompted for the password of user *pi*. If you have not changed it, it should be *raspberry*. When using Module sesman-Xvnc you get a new desktop. If you want to replicate the current desktop use Module console.



Tested remote desktop clients

VNC

Linux: Vinagre.

Windows: RealVNC Viewer, TightVNC.

Android: bVNC, RealVNC Viewer, VNC per Android.

IOS: RealVNC.

RDP

Linux: Vinagre.

Windows: Windows 10 Remote Desktop Client, Windows CE 5.0 Remote Desktop Client.

Android: load an old apk (microsoft-remote-desktop-8-0-5-24406-en-android.apk)

Auto Setup USB ports

On Linux serial USB ports are numbered in the sequence they are detected.

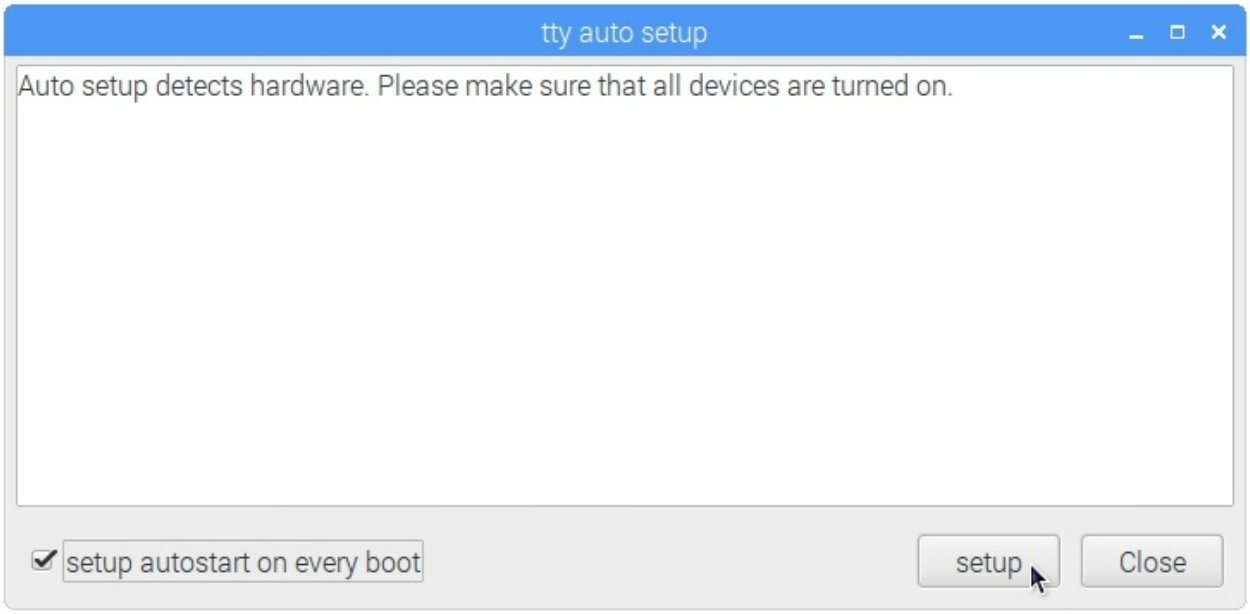
This could work some times. But it isn't reliable. If of any reason you loose a contact between the raspberry pi and an USB-serial port adapter. It could be that the entire configuration doesn't work anymore!

A more reliable way is, to give every serial port a name.

This is the reason why auto setup pop up on a fresh system.

Start tty auto setup

- all your devices should be connected to the internal \/ external USB-Hub (inclusive USB-gps-dongle and special CAN-BUS to USB adapter like actisense NGT-1 or !!link!!)
- turn on all your NMEA 0183 devices



- If setup autostart on every boot is checked uncheck it
- start the setup process by the setup buttom.

The tty auto setup can be started manually by Tools->Auto Setup Start

Auto setup is sometimes able to find

- NMEA 0183 ports

It auto detects the baud rate and checks if there are only NMEA 0183 sentences of one device-ID.

example1: $APxxx

- It will select the port name ttyOP_AP

example2: $GPxxx

- It will select the port name ttyOP_GP

If there are more device-IDs it will select ttyOP_MPX1, next tty_MPX2,...

If it detects an openplotter compatible CAN-USB adapter it selects ttyOP_N2K

To check if everything is done go to USB manager.

USB manager

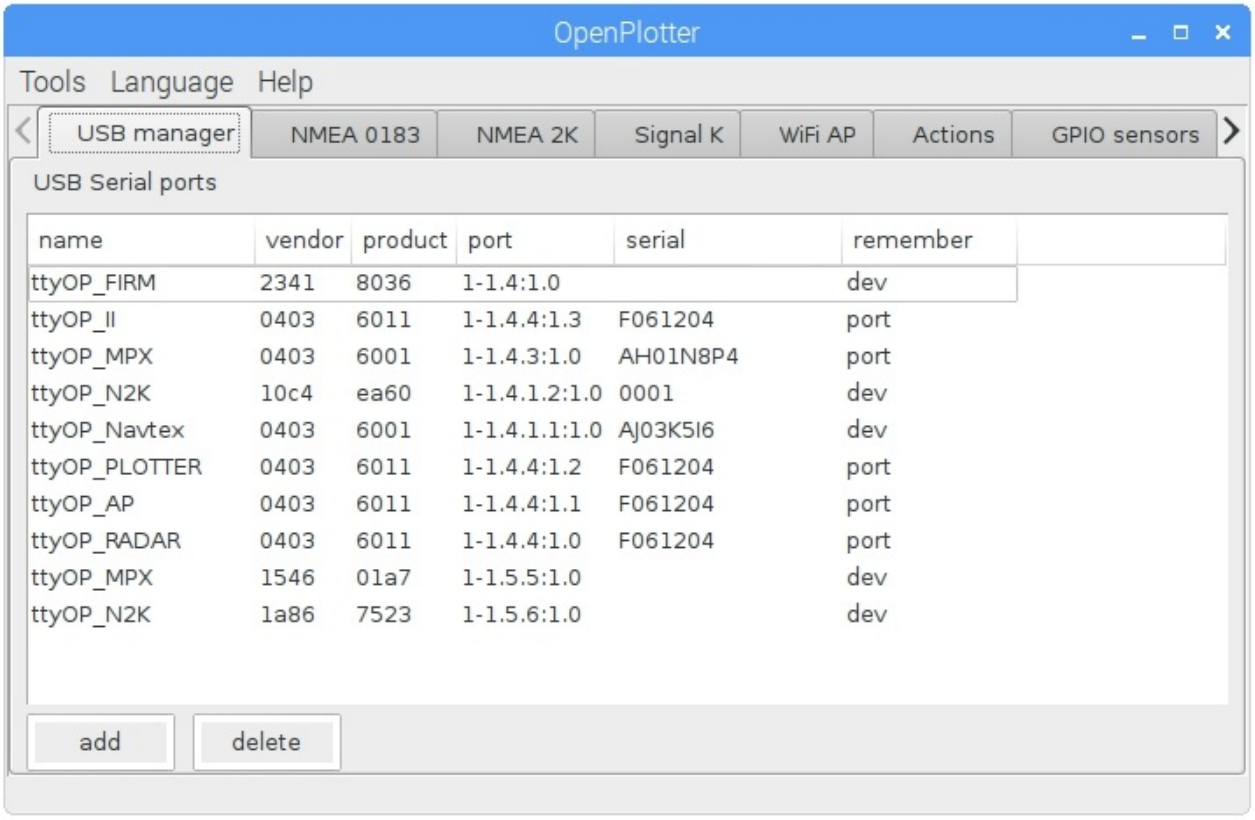
Why is it important to manage the serial ports?

Linux does use standard port names. They are named by the connection time.

If you can guarantee that every time all USB adapter are connected and that there is no chance of malfunction on one device and you never put another device to the system, then you don't need the USB manager.

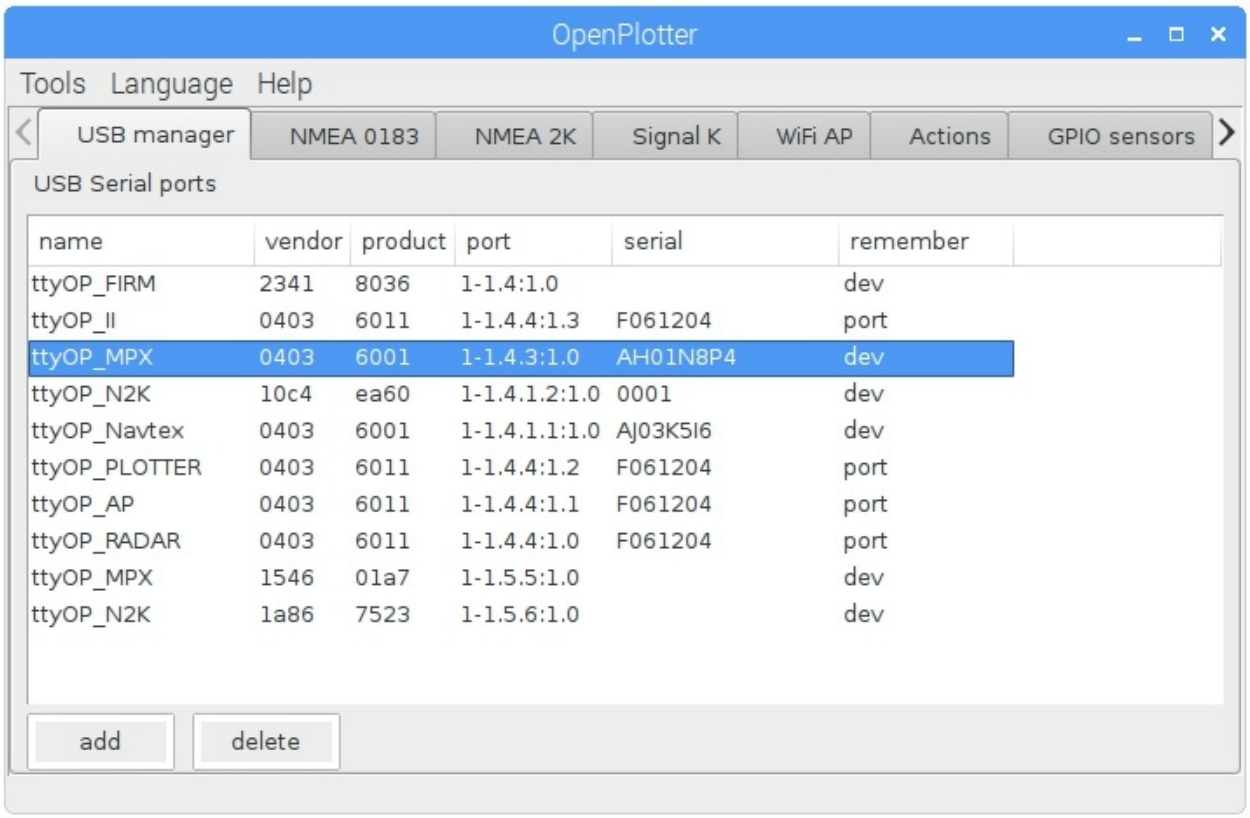
Otherwise the names ttyUSBx are changing (for example the AIS is on the autopilot port, the N2K Can-Bus is on the GPS port …). The multiplexer isn't able to do his job any more.

Here the USB-ports can be setup manually.



Picture 1: already configured USB ports

select add to open a window of not configured serial ports.

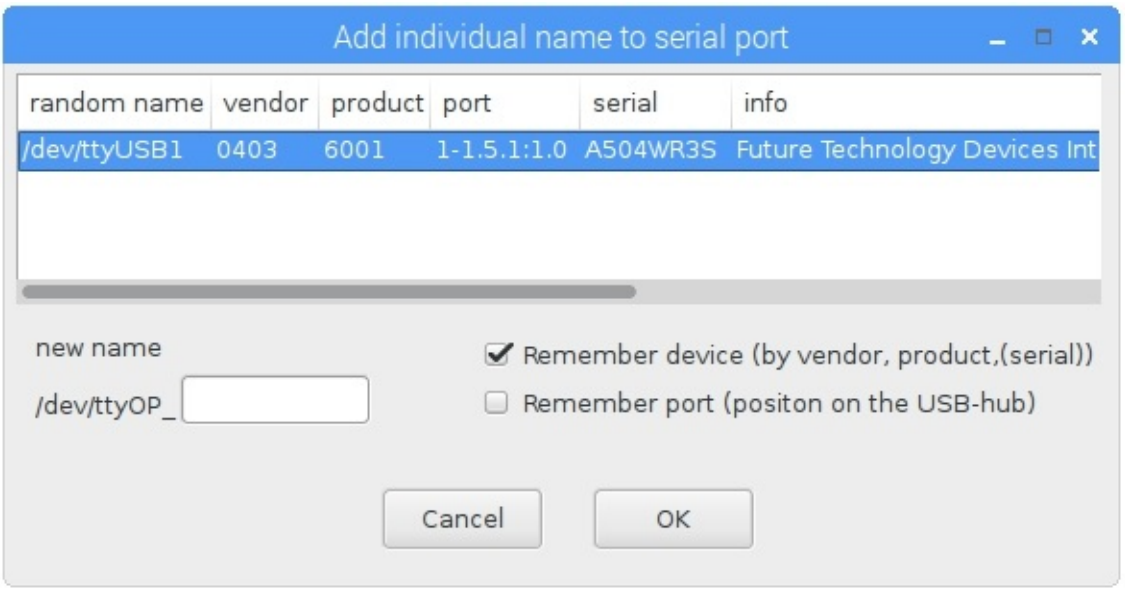


Picture 2: adding a FTDI port

random name: the numbered port name which is given by Linux

vendor product and serial: numbers and strings of the device

port: the USB port position



Picture 3: USB port names of the raspberry pi

If you add a new serial port you can select remember by device or remember by port.

Remember by port can be used every time. See picture 3 it shows the USB port names of the raspberry pi.

But normally you want the system to get the right name independent of the port it is connected to.

You can do this if there is not a second device with identical information. In picture 1 you see for example an 4 port FTDI hub. It has a vendor and a product number and a serial string. But you have 4 times identical data. So you can't use remember by device. There are cheaper adapter which haven't got a serial number. If you have more than 1 of them you also can't use remember by device.

Naming

The Linux names ttyUSB0, ttyUSB4 don't have any information what is connected to the port so it's better to use a meaningful name.

The USB manager uses ttyOP_xxxx.

Where xxxx belongs to personal decision. But we recommend to use

ttyOP_N2K: for actisense ngt-1

ttyOP_AP: for autopilot

ttyOP_GP: for GPS

ttyOP_II: for seatalk

ttyOP_FIRM: for arduino with firmata

ttyOP_MPXx: is used for multiplexers but it could be better to rename it to a meaningful name like AIS or PLOTTER.

There are other devices which doesn't send anything they're only listening for sentences add a meaningful name to them.

To change a port name double click the line.

Openplotter pages

- USB manager
- NMEA 0183 Multiplexer
- NMEA 2K
- SignalK
- WiFi AP
- Actions
- GPIO sensors
- I2C sensors
- 1W sensors
- SPI sensors
- Accounts
- MQTT
- SMS

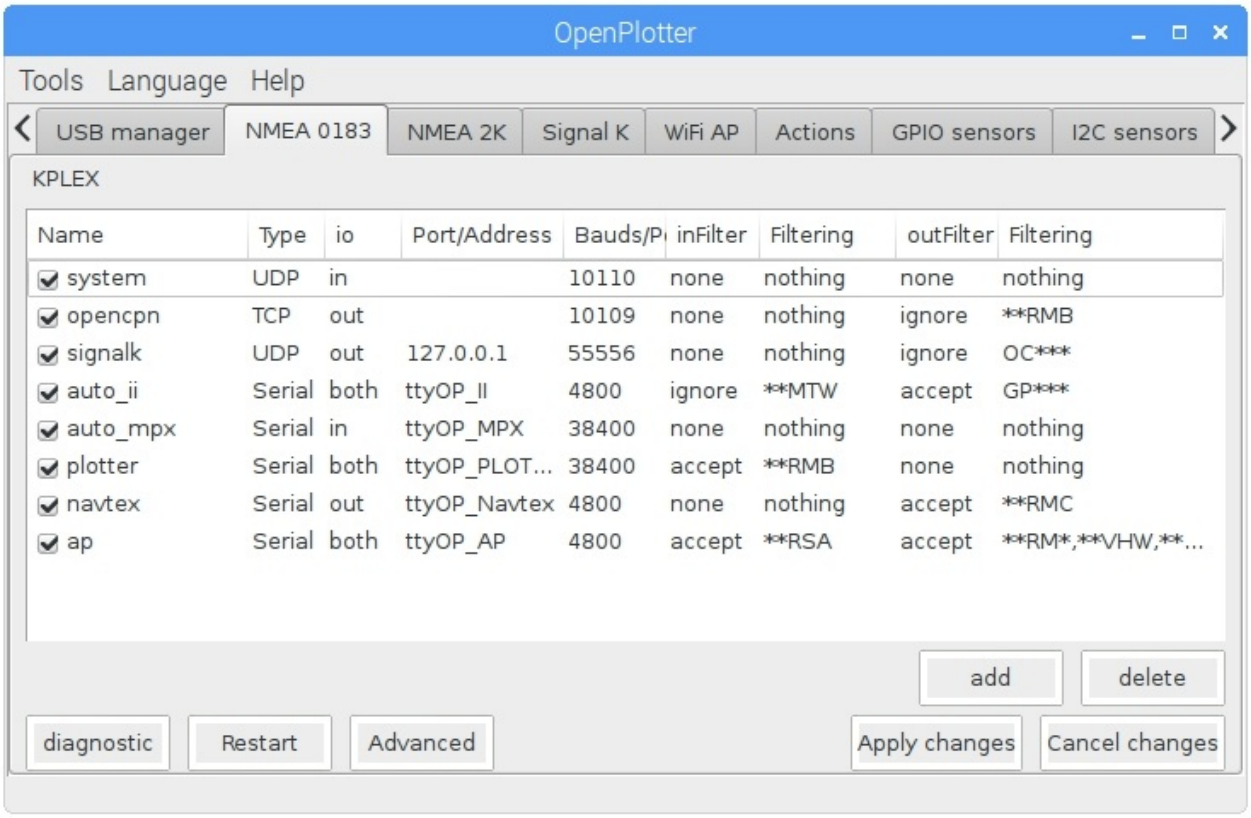
NMEA 0183 Multiplexer

It is important that you understand that OpenPlotter must drive all the NMEA data traffic to work properly, so you do not need to configure OpenCPN to get GPS signal, we will set this in OpenPlotter NMEA 0183 tab.

System defaults

There are three lines which couldn't be changed.

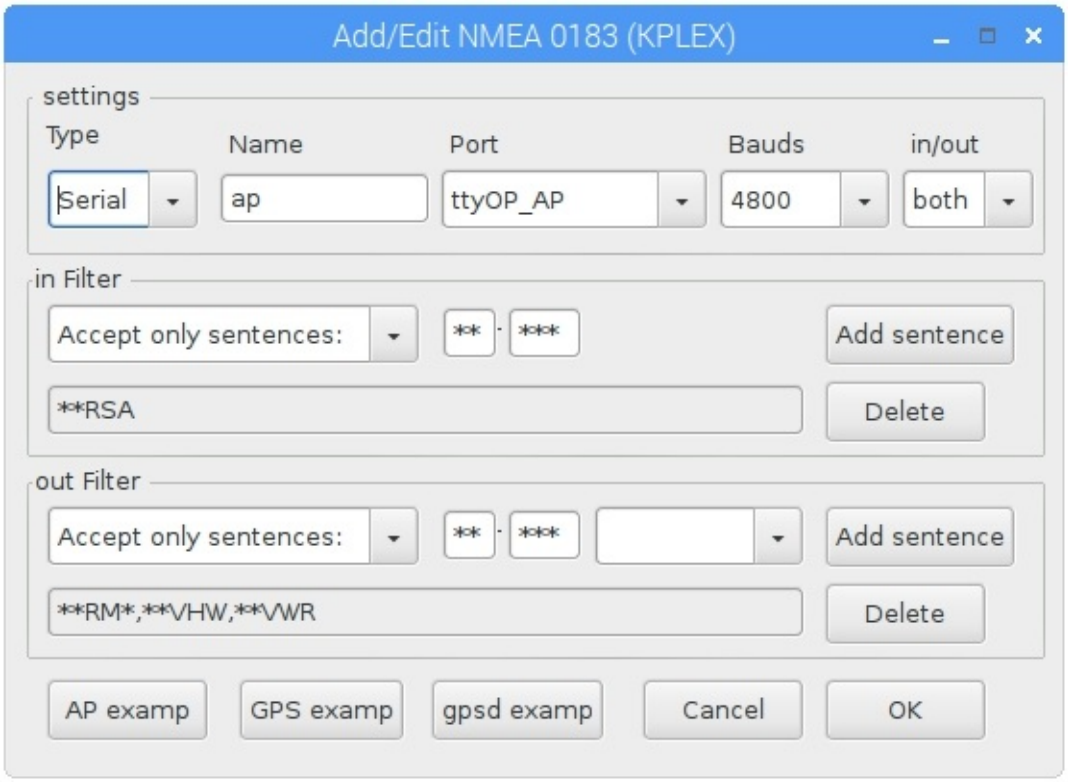
- system: Used by SDR-AIS, by Calculate and by NMEA 0183 generator
- opencpn: this is the datastream to opencpn
- signalk: this is the nmea 0183 stream to SignalK



Names with auto_xxx are made by Auto Setup USB ports to set the correct baudrate (the filters aren't set !!!).

Adding or changing a connection

- press add
- double click line to be edit



Kplex can work with Serial, TCP or UDP connection (Type).

Name: lower case letters should be meaningful

Port:

- Serial: The name of the port ttyxxxxx
- TCP / UDP: port number

Bauds: Baudrate for serial port

in / out:

- Serial: both, in, out
- TCP / UDP: in, out

in Filter: Data coming from the device

- none
- Accept only sentences
- Ignore sentences
    - **: device id
    - ***:type

  Use Add sentence to add a selection.

out Filter: Data send to the device. Same as above but filter can be connected to a given name

The three example buttons load default filter.

Diagnostics

On NMEA 0183 tab is the button diagnostic.

Select a line and click the button.

One or two windows pop up (two if setting is "both").

They show the transfered sentences

On the lower left side you can see the calculated transfer speed in baud.

! If the output speed is higher than the baudrate of the device some data will be cut. Select filters to send only needed data!

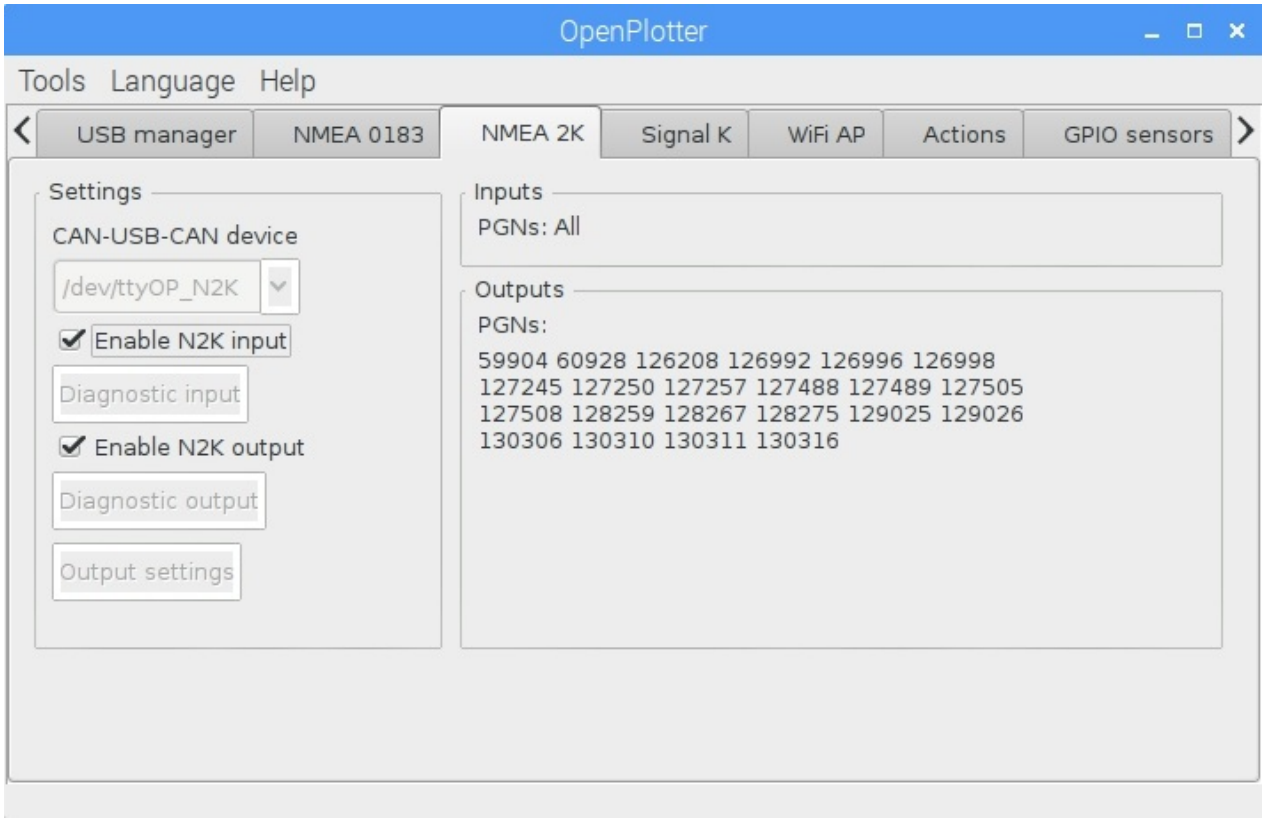
Advanced

There are many settings which could be done in OpenPlotter directly. But Kplex has advanced options which aren't used often. These settings or filter can be done between the lines

###end of OpenPlotter GUI settings

###Manual settings

NMEA 2K



Settings

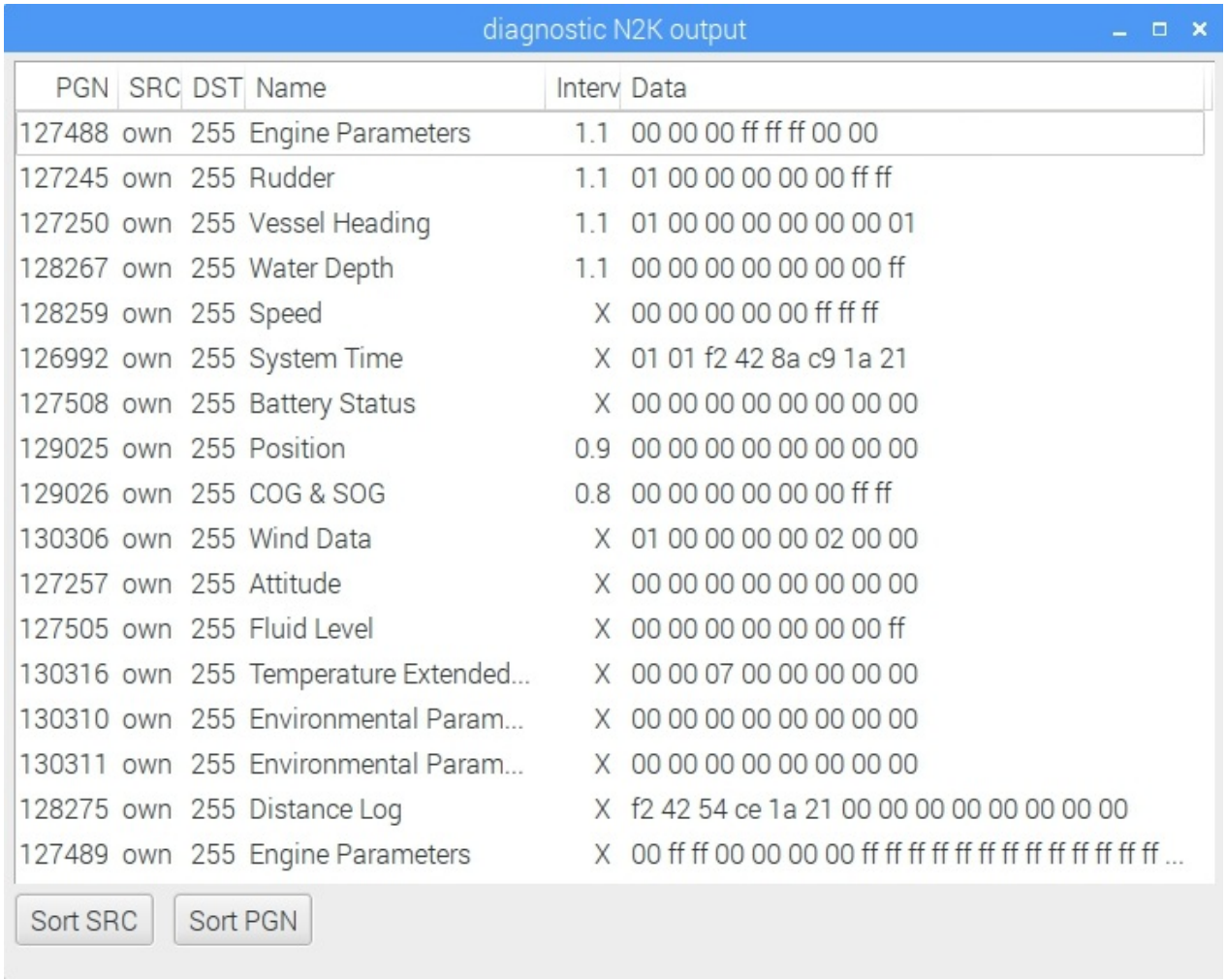
- select the N2K port
- activate Enable N2K input
- activate Enable N2K output

Diagnostic input

If button is gray, you have to deactivate Enable N2K input

Diagnostic output

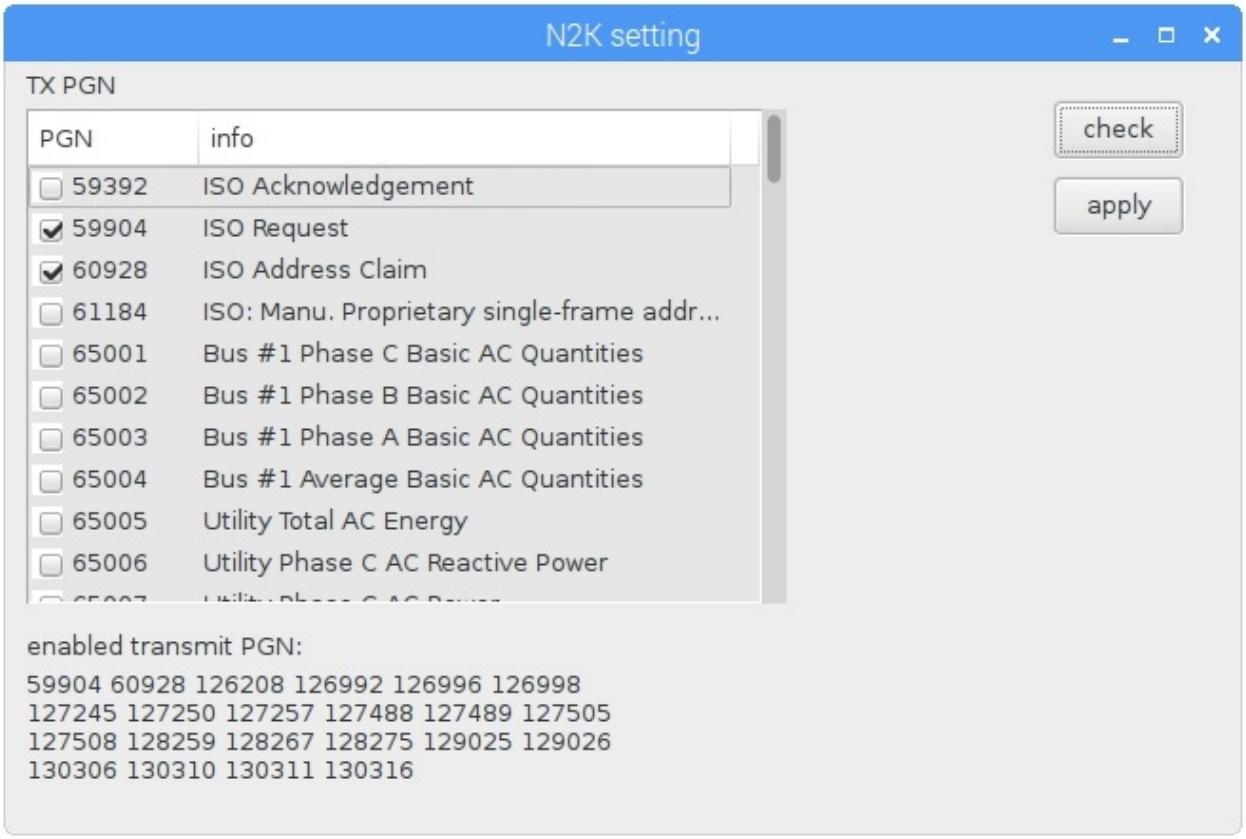
If button is gray, you have to deactivate Enable N2K output



Output settings

If button is gray, you have to deactivate Enable N2K input and Enable N2K output

These are hardware settings! If nothing is checked everything is blocked. You can't get anything out (security).

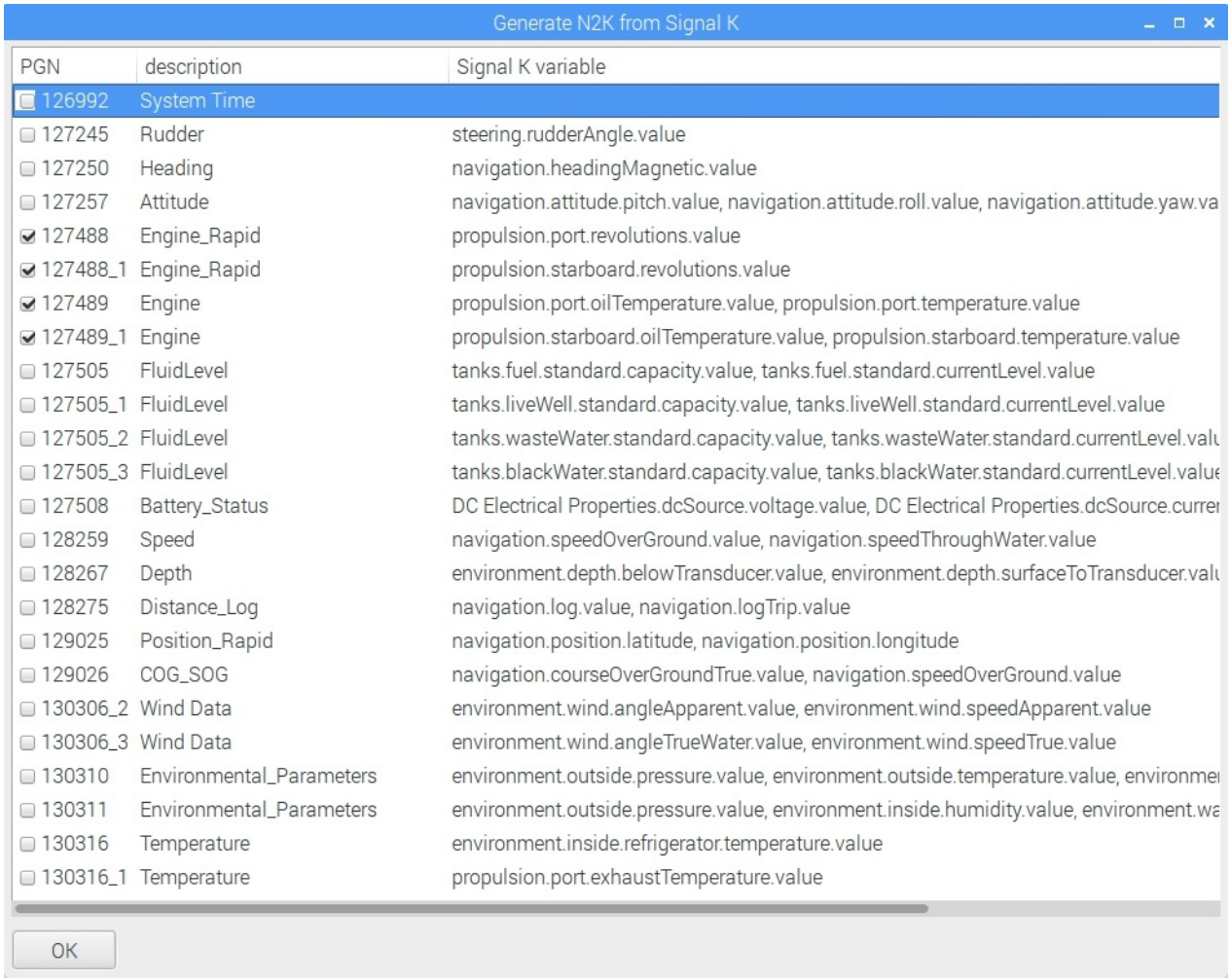


After opening the dialog box press two times check to read the settings from the CAN-BUS adapter.

First step is to enable needed PGNs for output. Then press apply and press check. Check if settings are done.

NMEA 2000 generator

Tools->NMEA 2000 generator



Check the PGNs you want to send onto the NMEA 2000 bus (it only works if the output settings for the PGNs are also set).

Signal K

Is the base protocol of OpenPlotter.

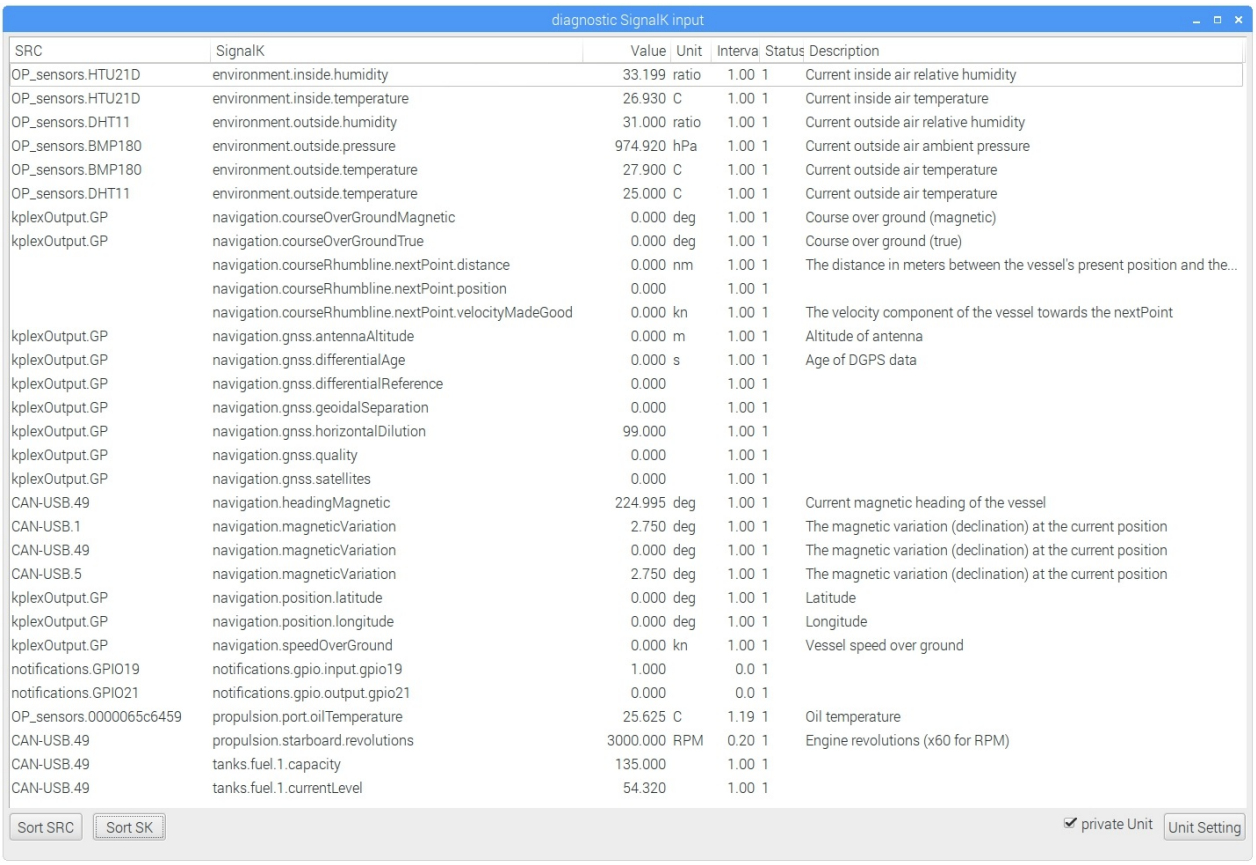
It is optimized for Internet browser. The data stream depend on json format.

both NMEA formats are converted to SK. The format is open and the values are readable. It uses strict SI-units.

On the SignalK tab you can enter the MMSI of the boat and "apply changes".

Diagnostic

SignalK diagnostic



SRC	SignalK	Value	Unit	Interva	Status	Description
OP_sensors.HTU21D	environment.inside.humidity	33.199	ratio	1.00	1	Current inside air relative humidity
OP_sensors.HTU21D	environment.inside.temperature	26.930	C	1.00	1	Current inside air temperature
OP_sensors.DHT11	environment.outside.humidity	31.000	ratio	1.00	1	Current outside air relative humidity
OP_sensors.BMP180	environment.outside.pressure	974.920	hPa	1.00	1	Current outside air ambient pressure
OP_sensors.BMP180	environment.outside.temperature	27.900	C	1.00	1	Current outside air temperature
OP_sensors.DHT11	environment.outside.temperature	25.000	C	1.00	1	Current outside air temperature
kplexOutput.GP	navigation.courseOverGroundMagnetic	0.000	deg	1.00	1	Course over ground (magnetic)
kplexOutput.GP	navigation.courseOverGroundTrue	0.000	deg	1.00	1	Course over ground (true)
	navigation.courseRhumbline.nextPoint.distance	0.000	nm	1.00	1	The distance in meters between the vessel's present position and the...
	navigation.courseRhumbline.nextPoint.position	0.000		1.00	1	
	navigation.courseRhumbline.nextPoint.velocityMadeGood	0.000	kn	1.00	1	The velocity component of the vessel towards the nextPoint
kplexOutput.GP	navigation.gnss.antennaAltitude	0.000	m	1.00	1	Altitude of antenna
kplexOutput.GP	navigation.gnss.differentialAge	0.000	s	1.00	1	Age of DGPS data
kplexOutput.GP	navigation.gnss.differentialReference	0.000		1.00	1	
kplexOutput.GP	navigation.gnss.geoidalSeparation	0.000		1.00	1	
kplexOutput.GP	navigation.gnss.horizontalDilution	99.000		1.00	1	
kplexOutput.GP	navigation.gnss.quality	0.000		1.00	1	
kplexOutput.GP	navigation.gnss.satellites	0.000		1.00	1	
CAN-USB.49	navigation.headingMagnetic	224.995	deg	1.00	1	Current magnetic heading of the vessel
CAN-USB.1	navigation.magneticVariation	2.750	deg	1.00	1	The magnetic variation (declination) at the current position
CAN-USB.49	navigation.magneticVariation	0.000	deg	1.00	1	The magnetic variation (declination) at the current position
CAN-USB.5	navigation.magneticVariation	2.750	deg	1.00	1	The magnetic variation (declination) at the current position
kplexOutput.GP	navigation.position.latitude	0.000	deg	1.00	1	Latitude
kplexOutput.GP	navigation.position.longitude	0.000	deg	1.00	1	Longitude
kplexOutput.GP	navigation.speedOverGround	0.000	kn	1.00	1	Vessel speed over ground
notifications.GPIO19	notifications.gpio.input.gpio19	1.000		0.0	1	
notifications.GPIO21	notifications.gpio.output.gpio21	0.000		0.0	1	
OP_sensors.0000065c6459	propulsion.port.oilTemperature	25.625	C	1.19	1	Oil temperature
CAN-USB.49	propulsion.starboard.revolutions	3000.000	RPM	0.20	1	Engine revolutions (x60 for RPM)
CAN-USB.49	tanks.fuel.1.capacity	135.000		1.00	1	
CAN-USB.49	tanks.fuel.1.currentLevel	54.320		1.00	1	

Data can be sorted by source or by SignalK.

To change the units click on Unit Setting

If private Unit is unchecked standard SI-units are shown.

SignalK Simulate

tools->SignalK Simulator

It's a tool which streams Data to the SignalK server. You can test it with Diagnostic.

There is a ".conf" file which is opened by the setting buttom

item_8 = [8, 'environment.wind.speedApparent', 4, 0, 40, 0.5144441, 0]

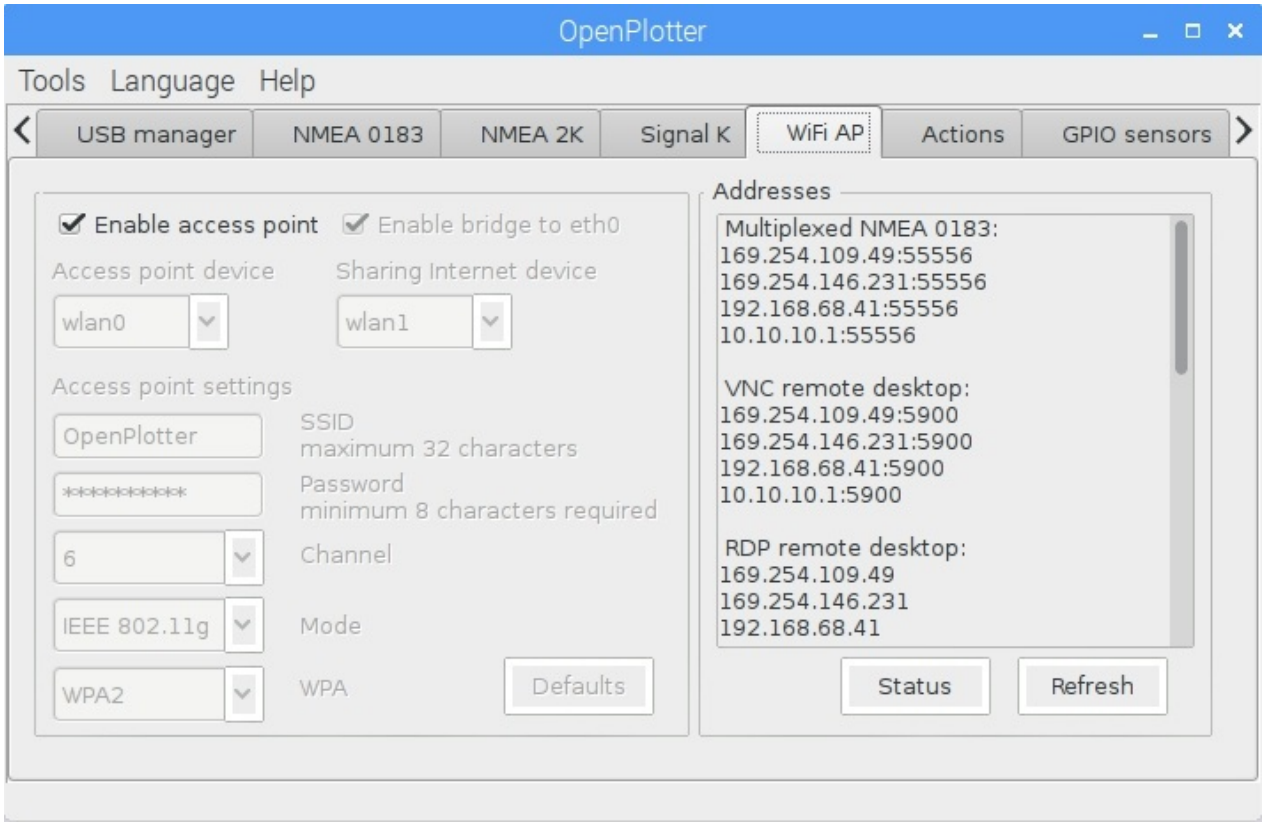
[Position in the GUI,SignalK-name,default value, min value, max value, multiplier, offset]

So it's easy to edit.

Start the GUI with the start buttom. The window is resizeable. Don't use more than 4 config lines on a small display.

!!! Never use this tool while sailing or boating !!!

WiFi AP



Picture: RPI active as AP with bridge to the Ethernet port. The internet connection comes over wlan1

If **Enable access point** is selected the AP should be active. To change settings you have to disable AP mode. **But be careful you will lose the connection to the RPI if you work headless.**

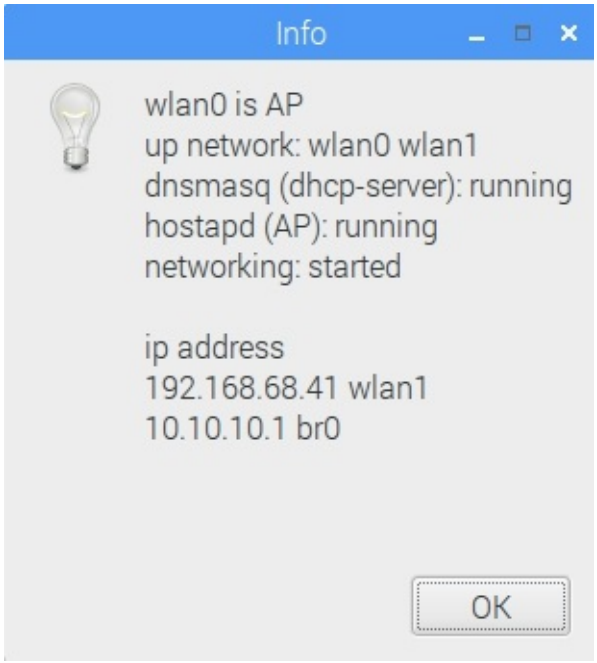
As **Access point device** you can only select AP mode able wlan (There were many changes in the RPI firmware and drivers only few wlan sticks work together. Sometimes if you add a non AP mode stick the AP mode for both is blocked.)

The second wlan is for connections to the internet (seaport wifi or mobile phone AP). If you do it this way all tablets mobile phones PCs haven't got to change the wlan when reaching a seaport. If the RPI has a connection to the internet (seaport wifi) all connected devices have internet access.

Sharing Internet can also be done by connecting to an internet router by Ethernet port. There is also a chances to connect to the internet with an gsm stick.(but it is easier to use mobile phone internet sharing over wifi)

The Ethernet port can be bridged to the AP to act like a normal router. This is important if you have a plotter with Ethernet port. Then the RPI acts like a gofree router.

When pressing the **Status** button



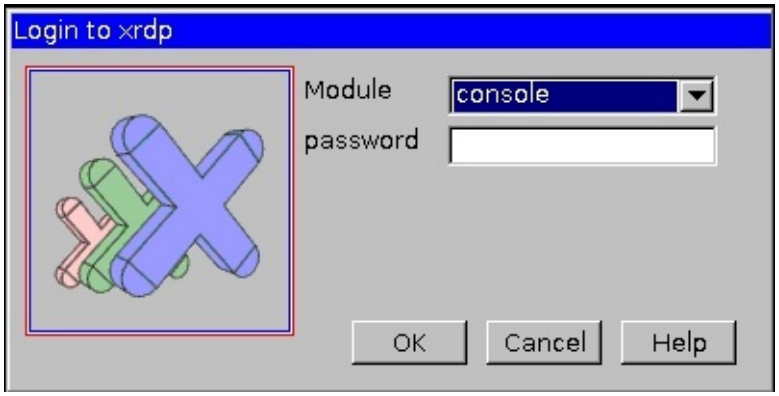
this window pops up and shows the network status.

RPI network manager

In the original RPI network-manager some devices show up as *device not managed*.

These devices are managed by OpenPlotter (in this example wlan0 is AP and bridge to eth0 is enabled).

In this example wlan1 can be managed by network-manager to connect to the internet.

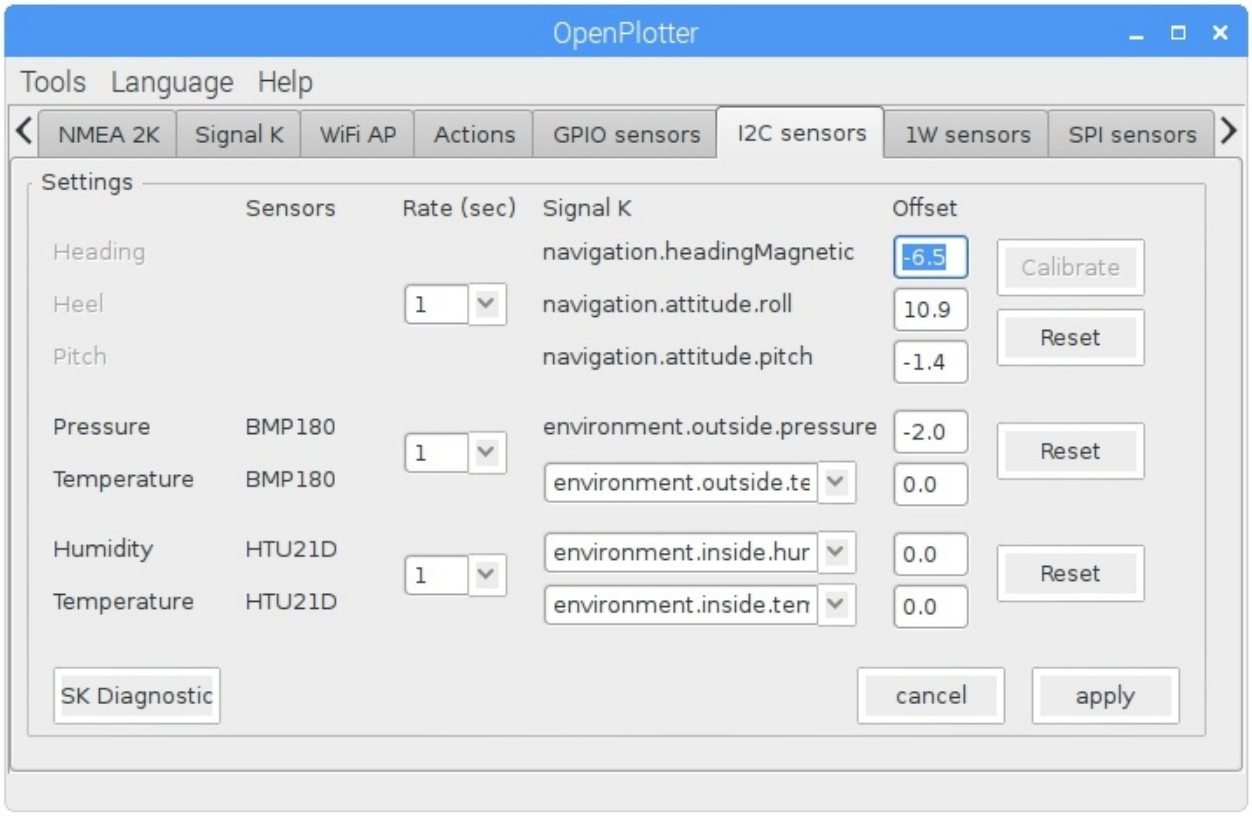


If you use remote desktop and want to connect to the internet by using the network-manager select **console**. Then you can connect to the internet otherwise linux security system blocks some settings.

GPIO sensors

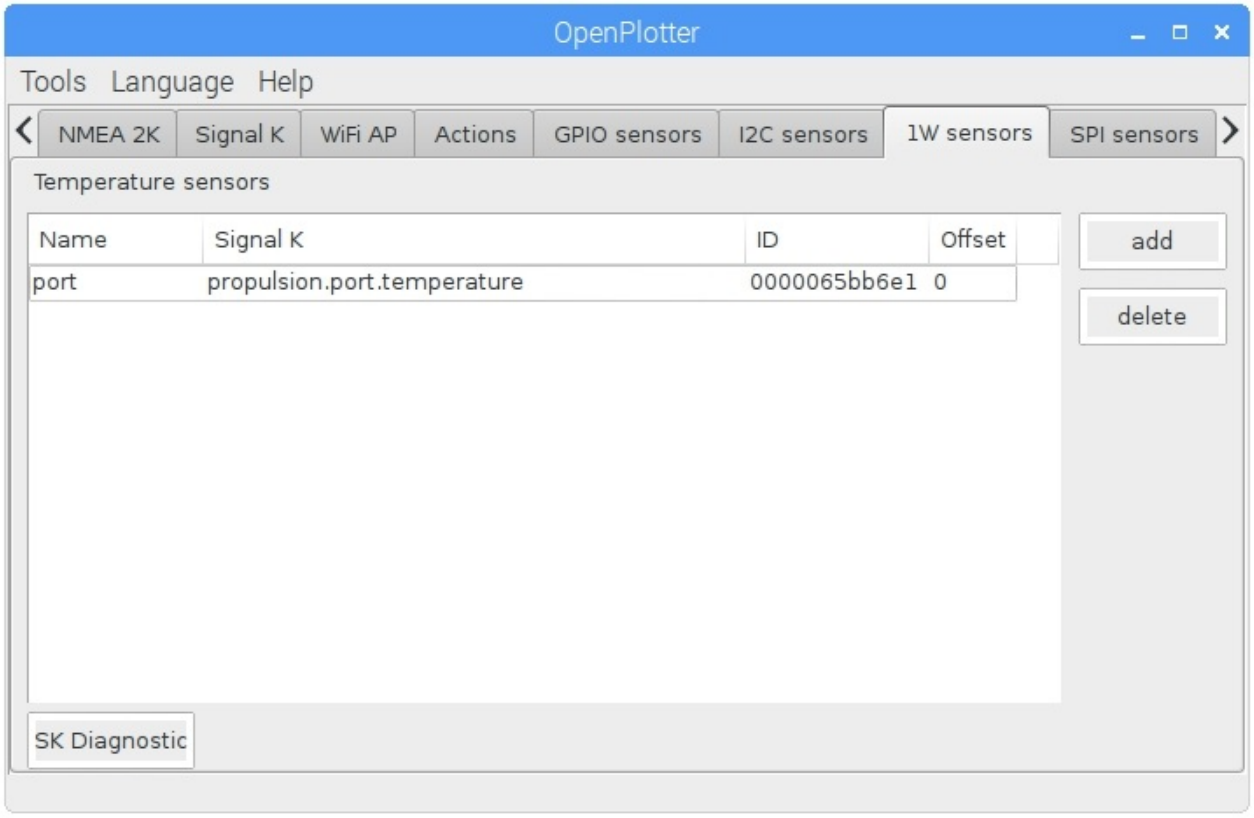
I2C

	Sensor
IMU	IMU Sensor
Pressure	Pressure/Temperature sensor
Temperature	
Humidity	Humidity/Temperature sensor

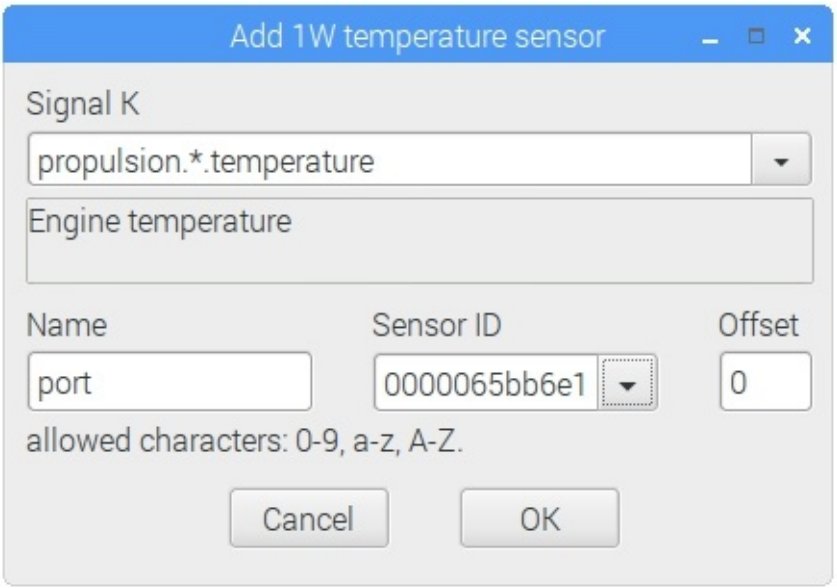


1W

- DS18B20
- One wire temperature sensor



Use add key to add another sensor. Double click the line to edit it.

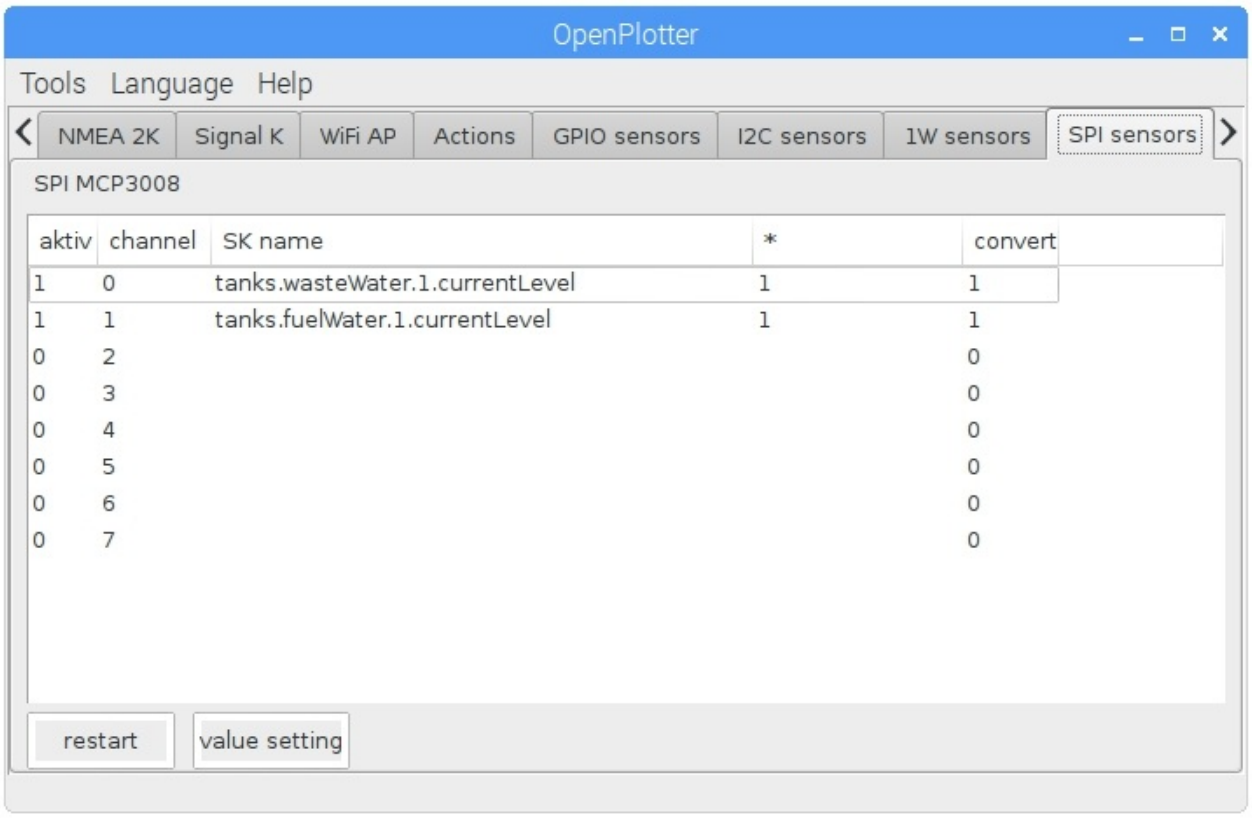


Use offset to correct value.

If there is a star in SignalK name. The text in field Name will replace the star.

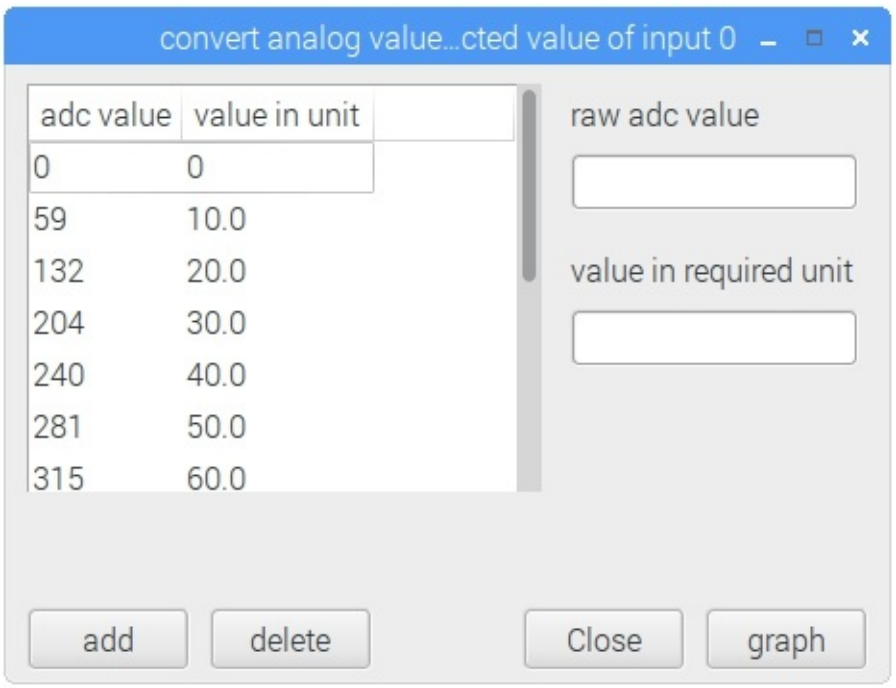
SPI

The adc (analog digital converter) MCP3008 is implemented in OpenPlotter.



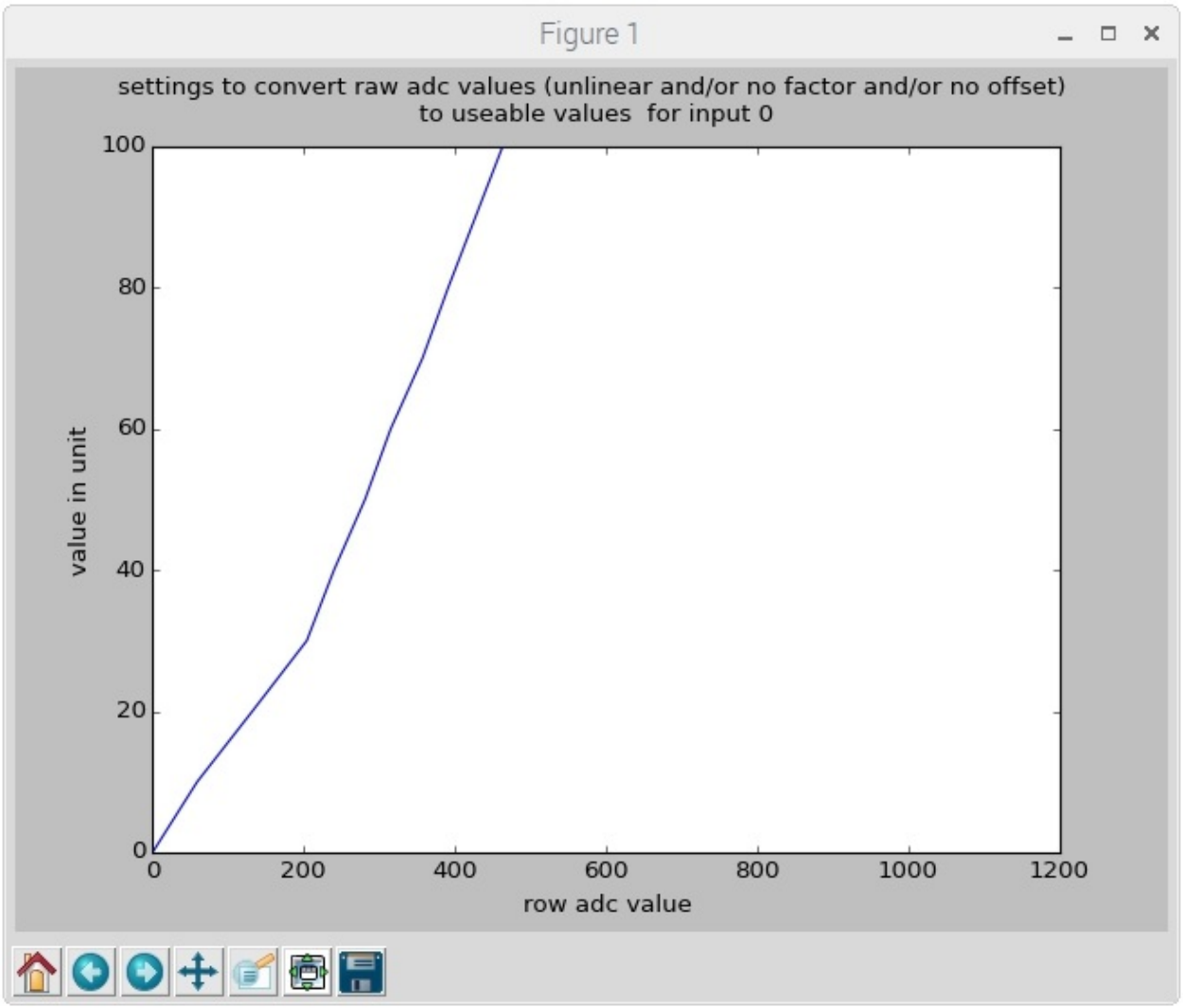
It does read voltage between 0 - 3.3 V. It is has a 10 bit resolution (0-1023).

Many signals aren't linear. To get a nearly linear result you can use the value setting.



The medium level in a tank on boats isn't linear to the height the sensor measures. To calibrate it empty it and insert the ADC value for 0%. fill in 10% of max volume and insert ADC value and 10.0. Go on until the tank is full.

Look at the graph you created by clicking on graph



Accounts

This chapter is under construction

To test remote monitoring by Twitter and Gmail feature, open new accounts. DO NOT use existing accounts or usual passwords because they can be exposed.

Open a new Gmail account and enable:

https://www.google.com/settings/security/lesssecureapps

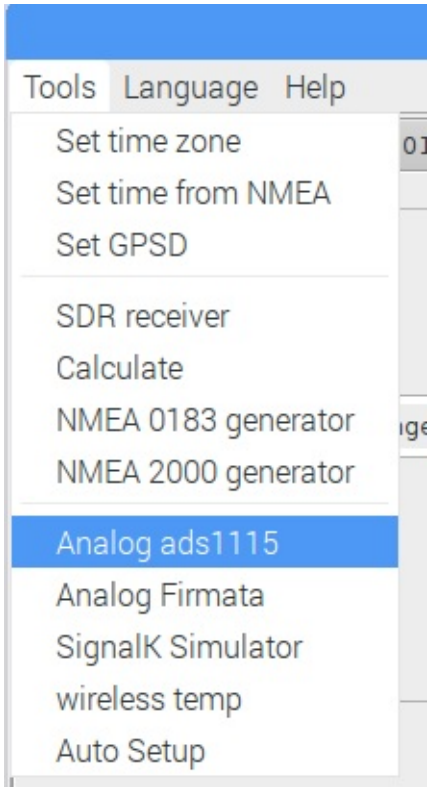
Using your new Gmail account, open a new Twitter account following this manual:

http://www.instructables.com/id/Raspberry-Pi-Twitterbot/step2/Create-ema

MQTT

Tools

In the tool menu are special modules.



The first three are:

- Set time zone

Opens a menu to choose a new time zone.

(It is the same as utc+x or utc-x)

- Set time from NMEA

The time from GPS (RMC-sentence) will be used to set the time of the RPI.

In future release it will read it from SignalK (then time can be taken not only from NMEA0183).

(If the RPI is connected to the internet it will get the time from a timeserver)

- Set GPSD

Opens the configuration file of GPSD in editor nano.

For details of GPSD google is your friend

- SDR receiver

SDR (Software Defined Radio) can be used to receive:

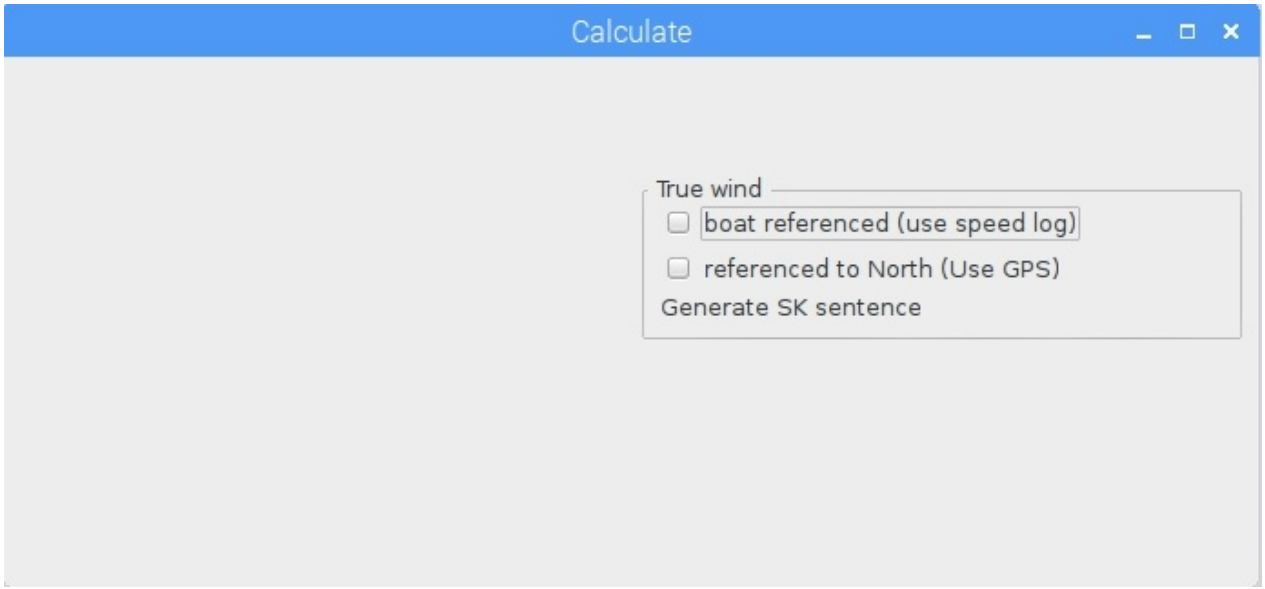
1. AIS
2. FM AM radio
3. VHF marine radio channel
4. weatherfax
5. 433/868 MHz remote controls or sensors
6. ...

Most cheap USB SDR can work from 30 Mhz - 1.7 GHz. So they can't receive navtex at 490 kHz or 518 kHz.

The antenna must be different for each frequency range to get good results.

- Calculate

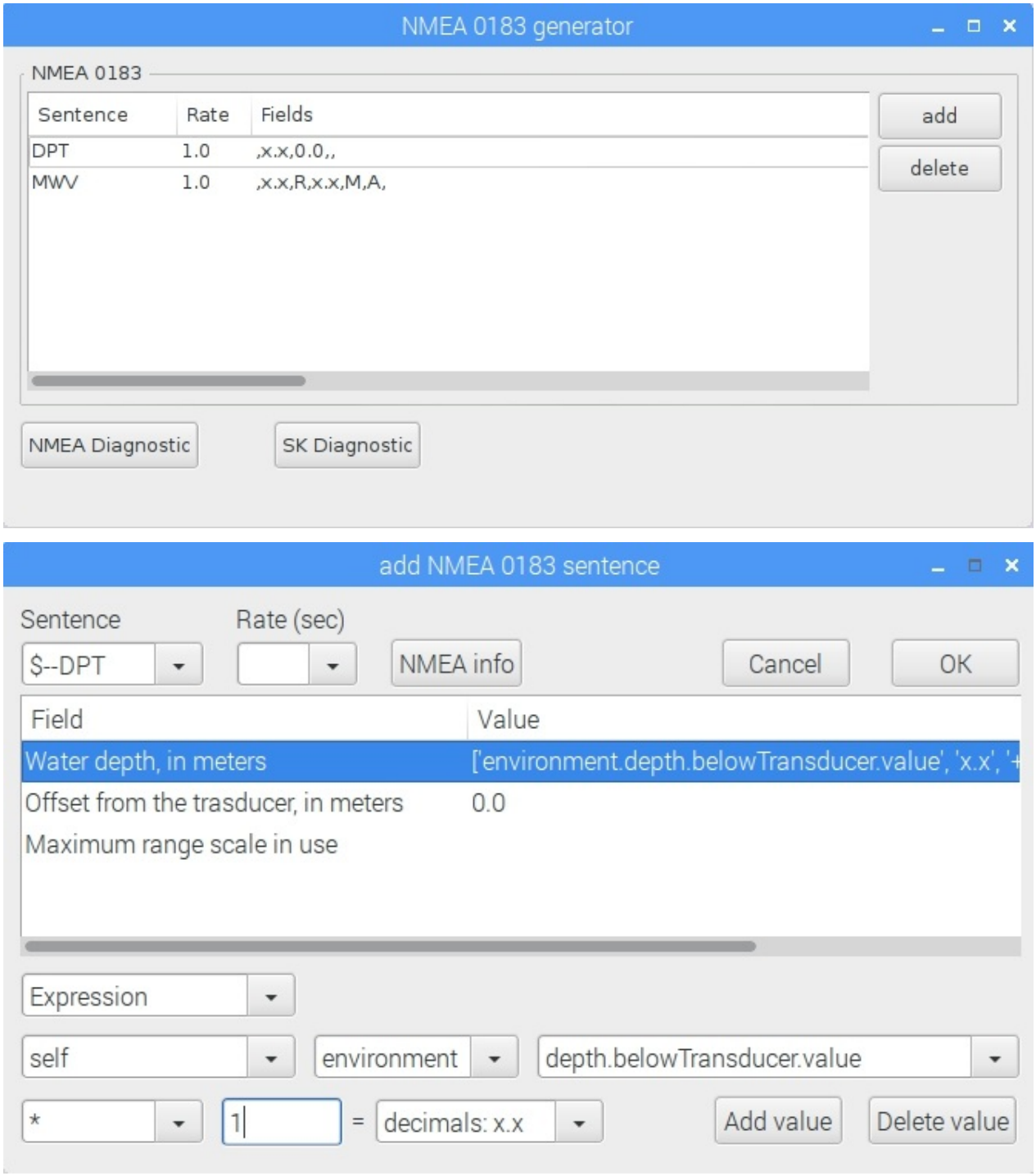
This is used to calculate the true wind from apparent wind



- NMEA 0183 generator

This tool allows you to create your own NMEA0183 sentences. You can select which SignalK value you want to take.

SignalK values are based on SI units (m,m²,m³,m/s,m³/s,J,rad,K,Pa,A,V,Hz,W) NMEA 0183 uses different units. That means you have to subtract for example 273.15 from a temperature value and to get radiant to degree you have to multiply by 57.29578.



To activate the changed configuration go to *SignalK* tab and click restart.

To check the result of the changes go to *NMEA 0183* tab -click on system -click on diagnostic.

- NMEA 2000 generator

- configureable tools

Tools defined

OpenPlotter has a chance to integrate programs.

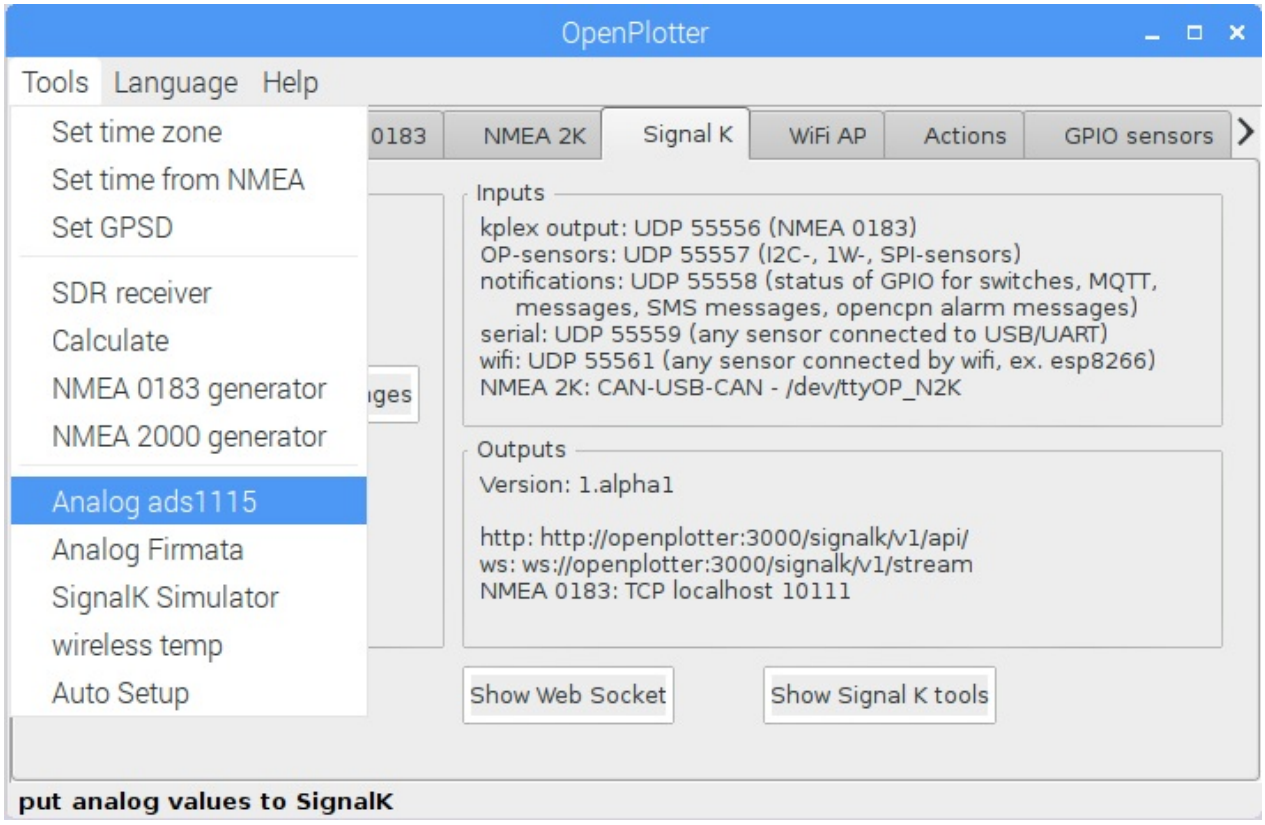
This was done to be open for add-ons done by experts and to keep OpenPlotter simple for newcomer.

In the **openplotter.conf** you can add or delete tools.

Example:

[TOOLS]

py = [['Analog ads1115', 'put analog values to SignalK', 'analog_ads1115.py', '0'], ['Analog Firmata', 'put analog values to SignalK', 'oppymata.py', '0'], ['SignalK Simulator', 'change values with sliders and send values to SignalK', 'SK-simulator.py', '0'], ['wireless temp', 'measure', 'rtl_433SK.py', '0'], ['Auto Setup', 'configure basic system', 'autosetup_tty.py', '0']]



The first name in the menu section is: *Analog ads1115*

The statusbar shows: *put analog values to SignalK*

The program to start is: *analog_ads1115.py*

The autostart on boot is disabled: *0*

So the first element of the list is *['Analog ads1115', 'put analog values to SignalK', 'analog_ads1115.py', '0']*

After menu selection this form opens.

settings - can be used to open a setup form or open a configuration file

start - normal start

stop - uses pkill command to stop the app

here some tools:

Analog ads1115

Analog Firmata

SignalK Simulator

wireless temp

Auto Setup

You can look at the code for these tools and build your own app/addon.

Analog ads1115

Is an adc with:

- 4 inputs
- 16 bit
- selctable gain
- selectable sample rate

When selected settings the configuration file will be opened.

[ADS1115_0] active = 1
gain = 0
samples = 6
ohmmeter = 0
fixed_resistor = 0
high_voltage = 3000
voltage_divider = 0
upper_resistance = 1000
lower_resistance = 1000
adjust = 0
sk_name = tanks.fuel.left.currentLevel
adjust_points = [[100.0,0.0],[5000.0,90.0],[9000.0,100.0]]

If a channel is used active must be set to 1

The gain setting can be changed with gain.

gain = 0 +/- 6.144V

gain = 1 +/- 4.096V

gain = 2 +/- 2.048V

gain = 3 +/- 1.024V

gain = 4 +/- 0.512V

gain = 5 +/- 0.256V

samples setting can be changed with samples (see table).

samples 0 8 samples per second

samples 1 16 samples per second

samples 2 32 samples per second

samples 3 64 samples per second

samples 4 128 samples per second

samples 5 250 samples per second

samples 6 475 samples per second

samples 7 860 samples per second

ohmmeter can be set to 1 to measure the resistance of a sensor.

fixed_resistor must be set to the resistance of the fixed resistor and

high_voltage is the voltage you put on the two resistors in serial.

The resistance you want to measure is connected on one side to ground and

on the other side to the fixed resistor and analog input of the adc.

The other side of the fixed resistor is connected to + pol (for example 3.3V (high_voltage))

voltage_divider can be set to 1 if you want to have Volt values.

If you want to measure 12 V with a adc witch is only capable of max 3.3V,

you need a voltage divider in form of two resistors. (50k+10k would put the voltage from 12V to 2V)

upper_resistance has to be set to 50000 and lower resistance to 10000 for this example.

Look at the picture on http://forum.arduino.cc/index.php?topic=214930.0

adjust is only for adjusting the offset

The value is send to SignalK on the name you set in sk_name.

adjust_points

You can put in some points of a curve.

Between this point the value is calculated linear.

This is the easiest way to get good results.

It can be used in combination with voltage_devider or ohmmeter.

Analog Firmata

An Arduino, Teensy or STM32 boards which can be programed by Arduino IDE or which have a library for standard firmata are useable.

The communication is done with USB port or an USB to serial converter. We recommend to use an isolator when using analog signals (usb to usb isolator or uart to USB isolator or use an ADUM1201 and a simple uart to serial converter).

When firmata standard is installed. Ad the new USB port in USB manager. **The name of the port has to be ttyOP-FIRM!**

Next step is edditing the configuration file.

Analog Firmata can be started. Best way to see if connecting works is to start openplotter from a terminal session.

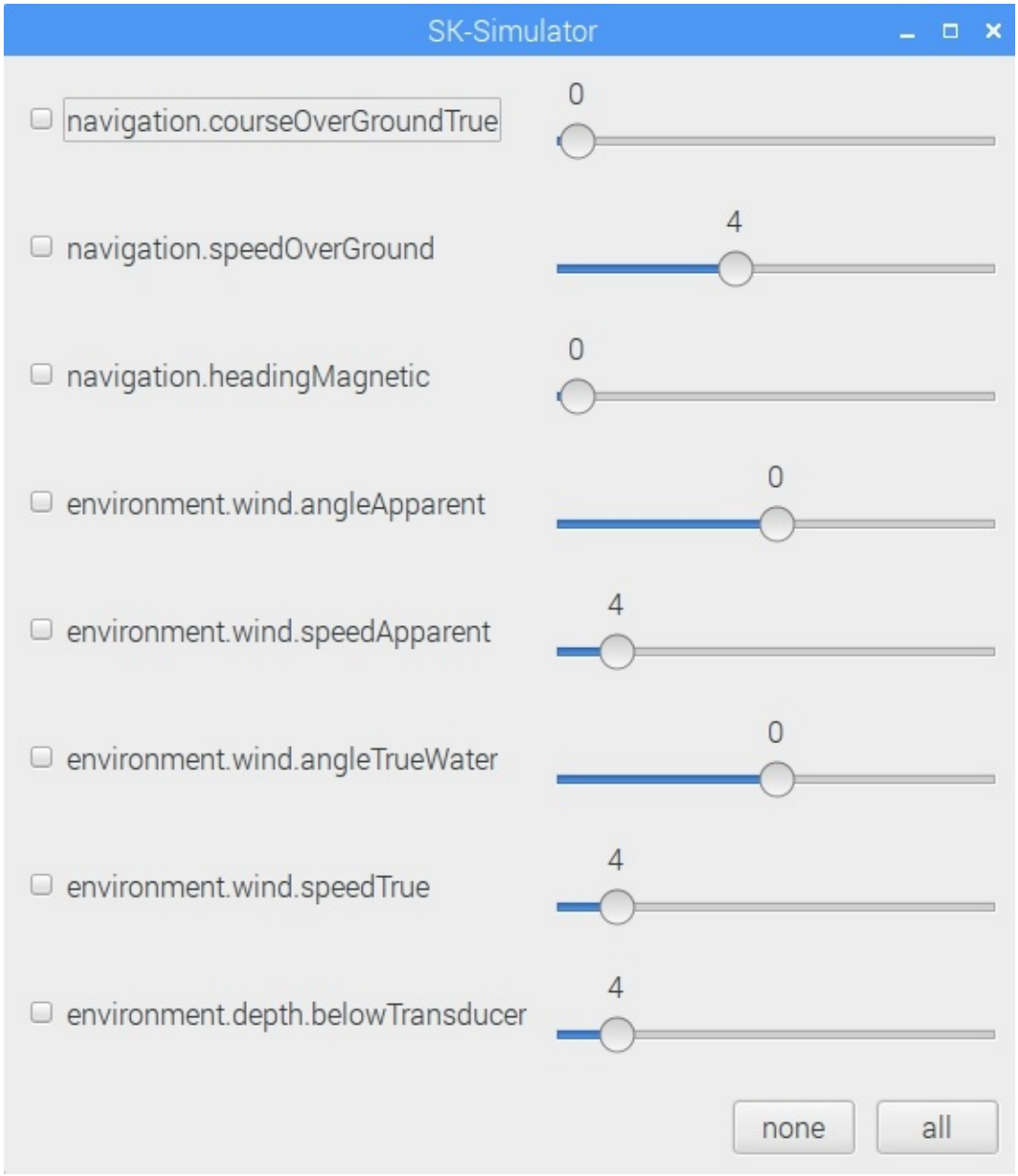
The start of pymata takes some seconds.

The configuration file is the same one which is used by ads1115.
There are up to 16 sections ([FIRMATA_0]..[FIRMATA_15])

Example:

[FIRMATA_1]
sk_name = tanks.fuel.right.currentLevel
adjust_points = [[52.0,0.0],[522.0,25.0],[708.0,50.0],[913.0,100],[1024.0,100.01]]

SignalK Simulator



Select the values you want. You can see your simulated values on SignalK **Diagnostic**.

You can use other SignalK values by editing the file _SK-simulator.conf. _It can be accessed by the setting buttom.

[main]

item_0 = [0, 'navigation.courseOverGroundTrue', 0, 0, 360, 0.0174533, 0]

item_1 = [1, 'navigation.speedOverGround', 4, 0, 10, 0.5144441, 0]

item_2 = [2, 'navigation.headingMagnetic', 0, 0, 360, 0.0174533, 0]

item_3 = [3, 'environment.wind.angleApparent', 0, -180, 180, 0.0174533, 0]

item_4 = [4, 'environment.wind.speedApparent', 4, 0, 40, 0.5144441, 0]

item_5 = [5, 'environment.wind.angleTrueWater', 0, -180, 180, 0.0174533, 0]

item_6 = [6, 'environment.wind.speedTrue', 4, 0, 40, 0.5144441, 0]

item_7 = [7, 'environment.depth.belowTransducer', 4, 0, 40, 1, 0]

The numbers behind the SignalK string copied from first line (*0, 0, 360, 0.0174533, 0)*

are:

- default value: 0
- minimum value: 0
- maximum value: 360
- factor: 0.0174533 (PI/180 for converting deg to rad)
- offset: 0 (interesting for Fahrenheit)

under construction

SDR receiver

DVB-T dongle (AIS)

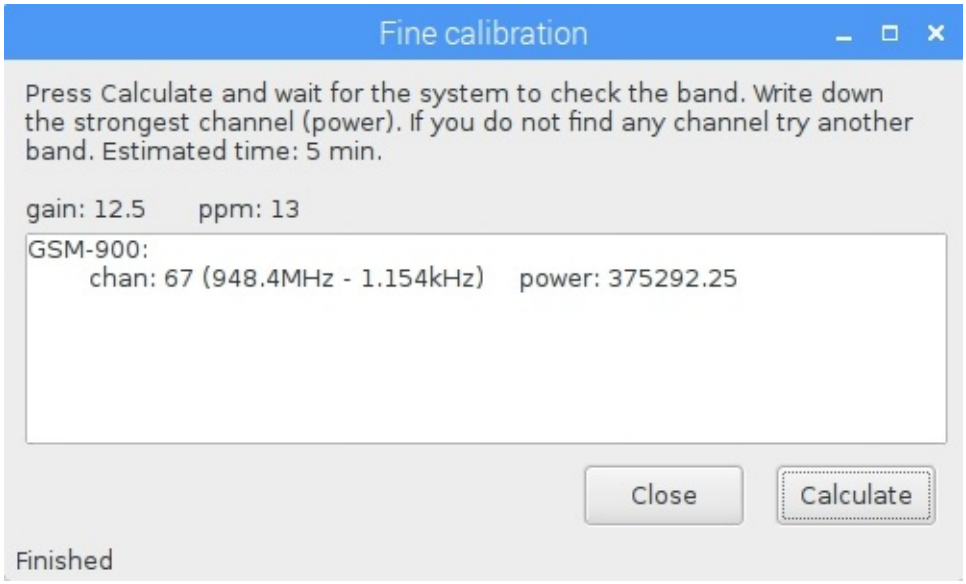
DVB-T dongles based on the Realtek RTL2832U chip and the new R820T2 tuner can work as a SDR AIS receiver.

A DVB-T dongle will need more power than the Raspberry Pi USB port can provide. You need to plug the dongle into a powered USB hub. Connecting and disconnecting can draw too much power and cause malfunction, try to do it when the system is off.

OpenPlotter is ready to get SDR AIS signal out of the box, you just have to calibrate to find **gain** and correction (**ppm**) values.

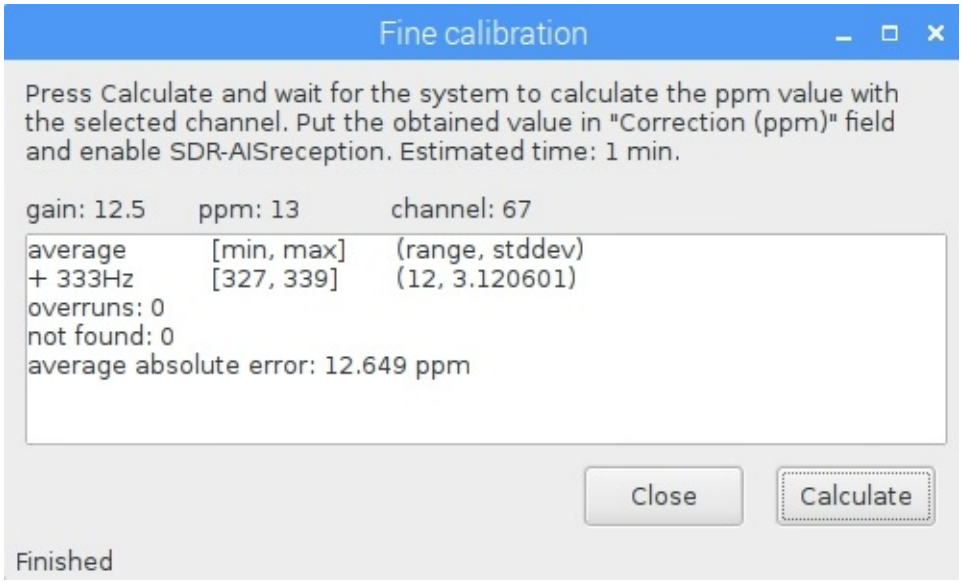
Fine calibration

It takes a few minutes after starting **Check band** and push **Calculate**.



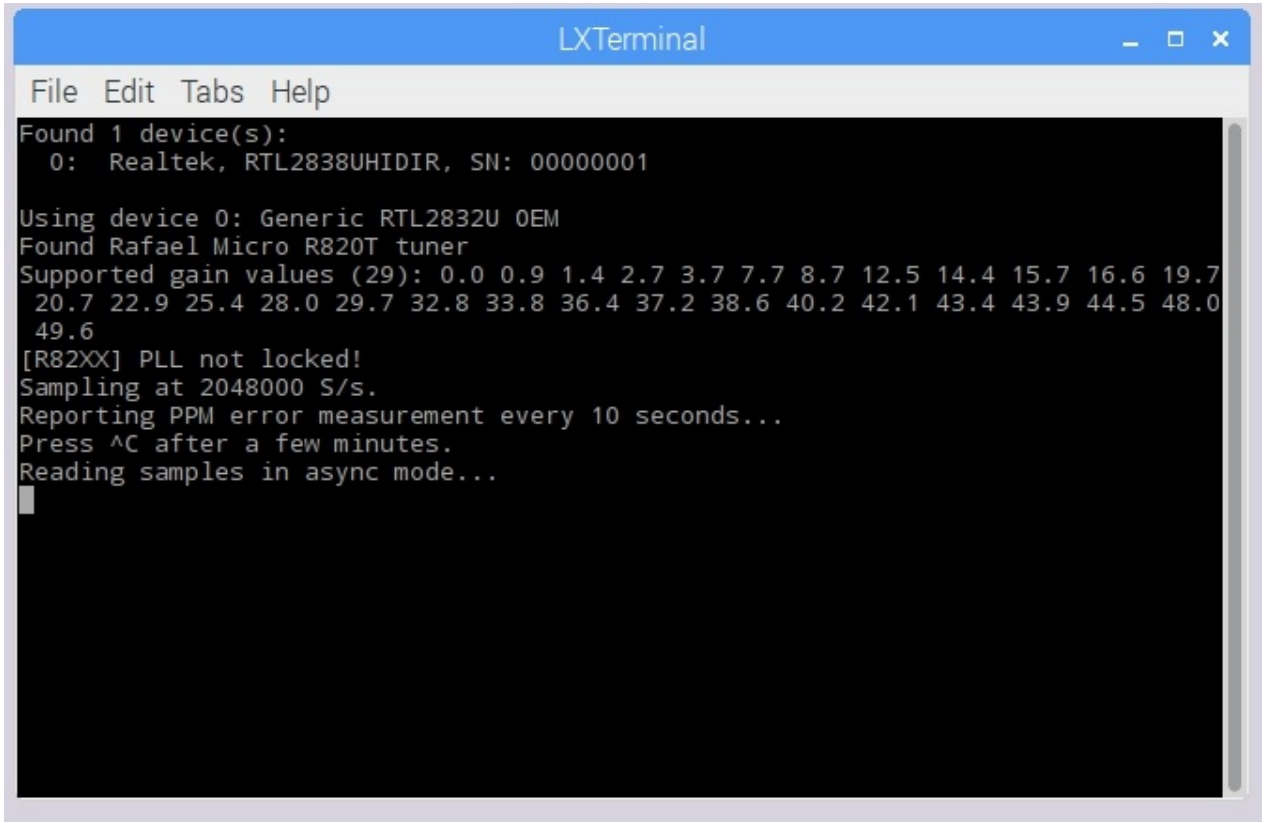
This is an example result.

If there are more "chan:" take the line with the highest power and put the "chan:" into **Channel** field and push **Fine calibration** and then **Calculate.**



For this example the field **Correction (ppm)** should be set to 13.

Now we look for the gain setting. Please push buttom **Calibration**.



The line **Supported gain values** is interesting for us. The max gain here is 49.6. Input this into the **Gain** field.

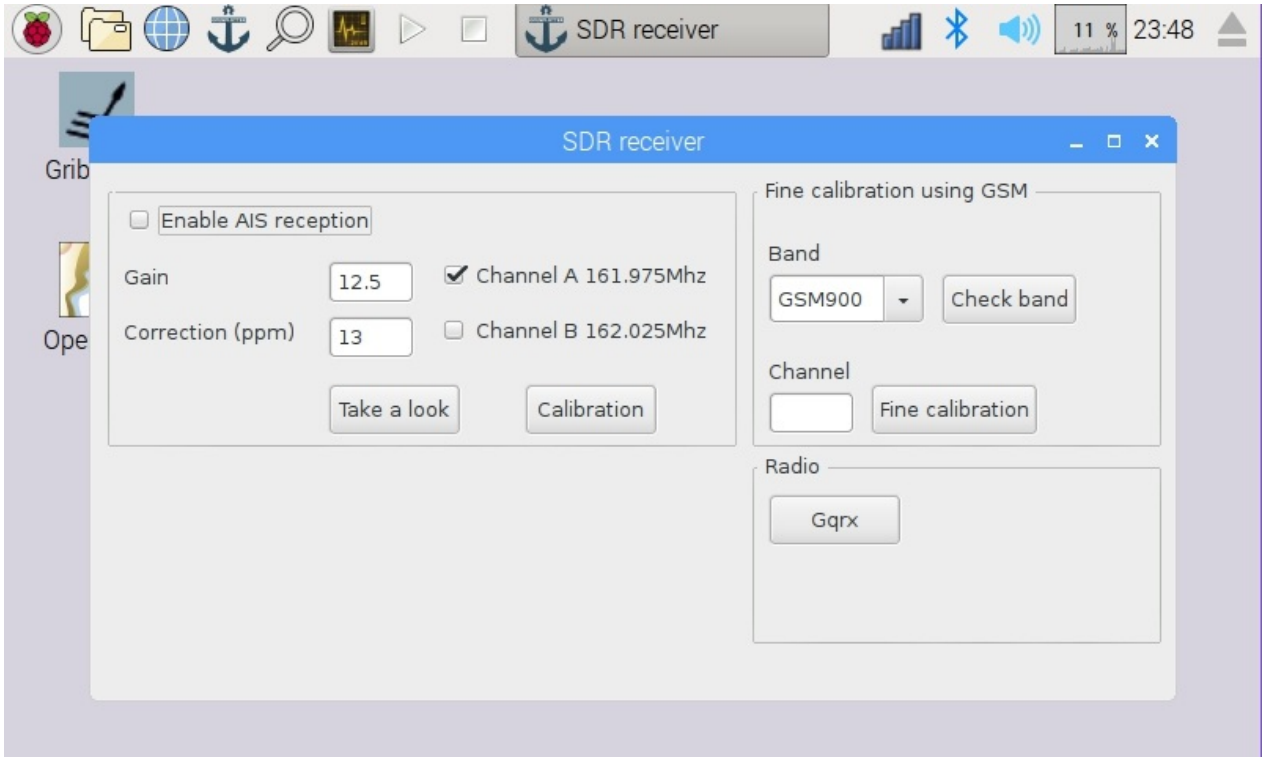
You can buy our DVB-T dongle and we can calibrate it for you and include a note with the gain and ppm values:

http://shop.sailoog.com/en/4-usb-sdr-ais-receiver.html

or you can follow this detailed guide:

http://sailoog.dozuki.com/Guide/Connecting+and+calibrating+SDR-AIS+dongles/3

AIS - receiving



Once you have found your **gain** and **ppm** value, select ***Enable AIS reception***.

You do not need to enable the rtlsdr plugin in OpenCPN. If you want to use that plugin you must disable SDR AIS reception in OpenPlotter.

Antenna

Although you can get to receive some boat with the supplied mini antenna, it is not enough for optimal reception of AIS frequencies. Any VHF antenna with the appropriate connector adapter will work fine. The antenna connector type of the dongle is female MCX.

Some home-made antennas:

http://sdrformariners.blogspot.com.es/p/blog-page.html

http://nmearouter.com/docs/ais/aerial.html

https://www.youtube.com/watch?v=SdEglNHyHB4

gqrx

gqrx is an application which also use the rtl2832u to receive something.

This gqrx-2.6-rpi3-1 version only supports RPI 3!

You can listen to:

- Radio in different waves
- vhf radio
- data

Be sure that the rtl2832u isn't used by openplotter or any other application. **It does consume much processor power. Other processes could be negative affected.**

Settings to begin with:

- configure I/O devices Input rate 240000
- Receiver Option Mode WFM (mono) FM radio
- Receiver Option Narrow FM FM marine radio
- FFT Settings Rate 5 fps 0 fps for low processor load
- Audio Gain 24.8 dB

wireless temperature sensors

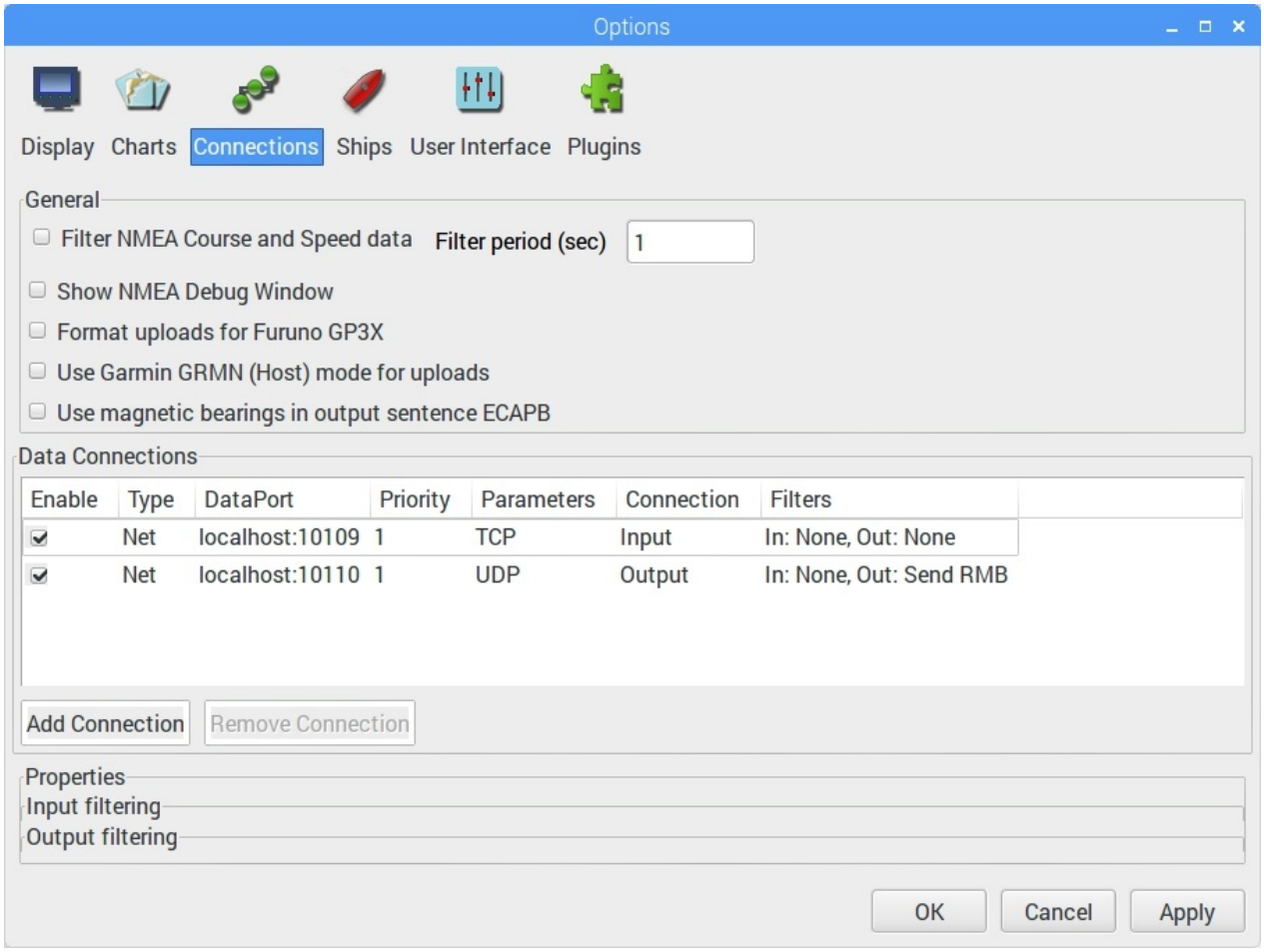
advanced alpha features for experts

433 MHz temperature and humidity sensors known from wireless home weatherstations can be connected via rtl2832u to openplotter.

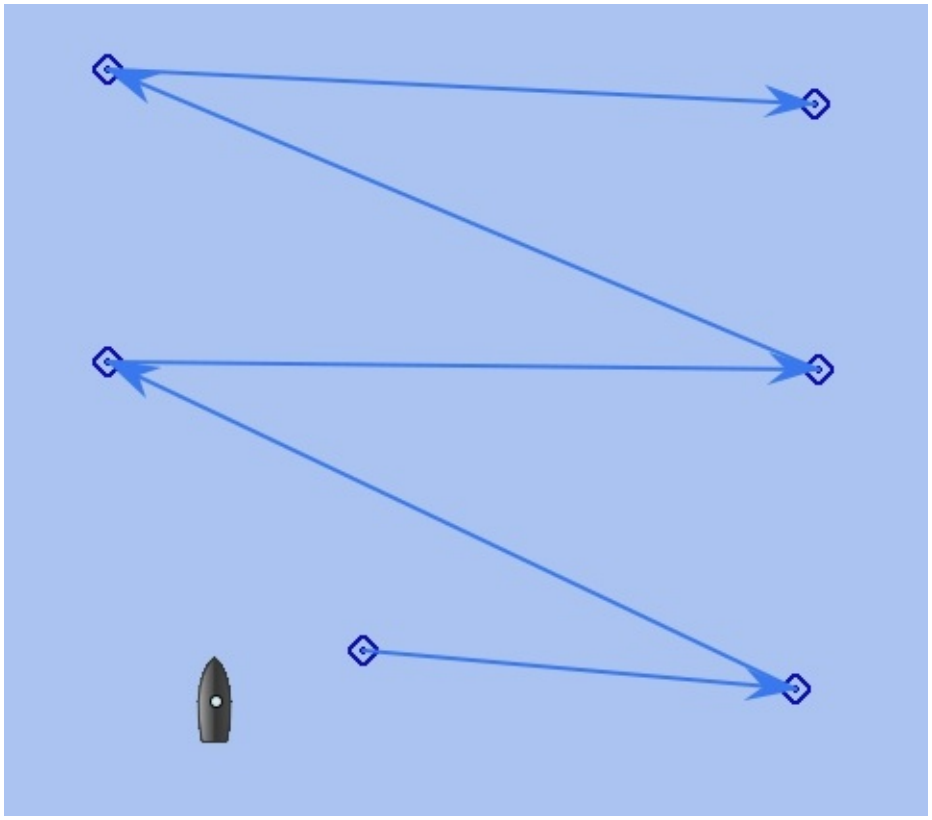
To translate this sensors to signalk use the python script tools\/rtl_433SK.py.

It can be integrated as tool. Add "[['433 Temp', 'get wireless temp', 'rtl_433SK.py', '0']]" this behind "py=" in openplotter.conf section [TOOLS].

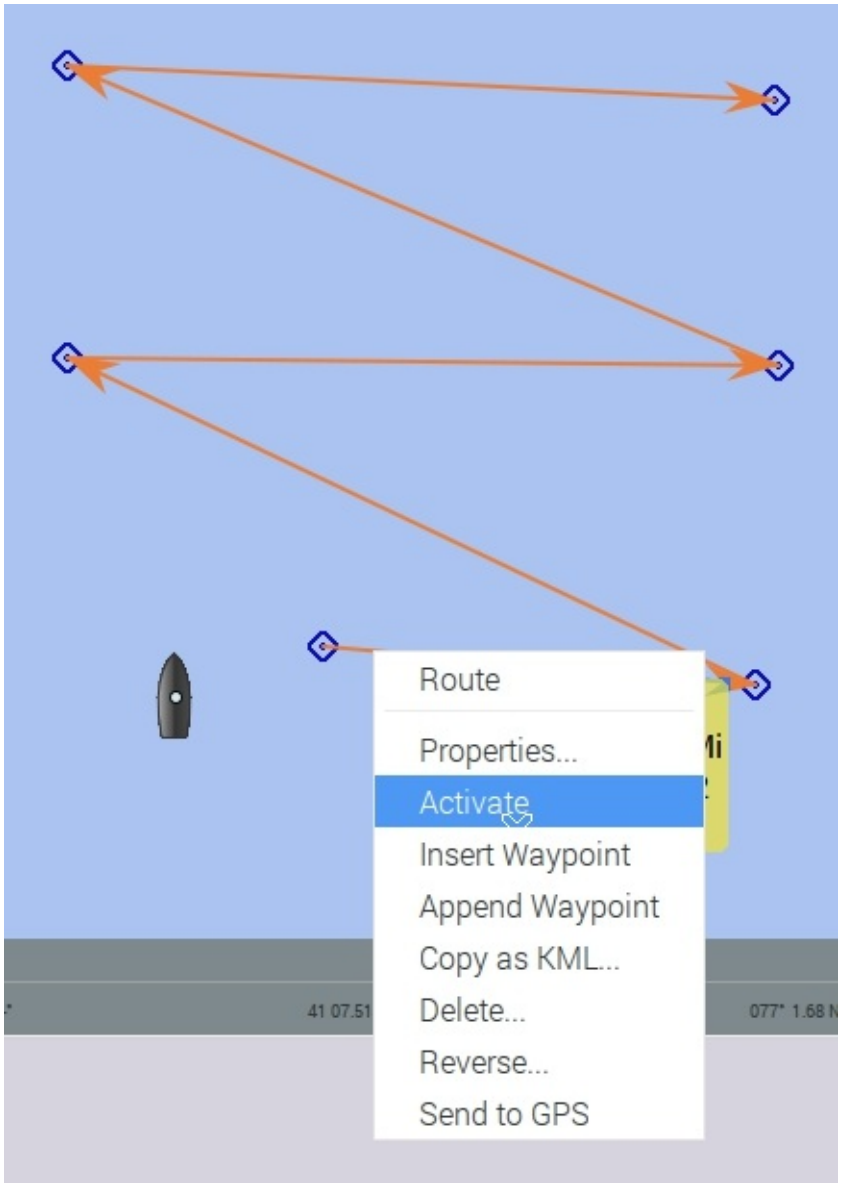
Chartplotter (OpenCPN)

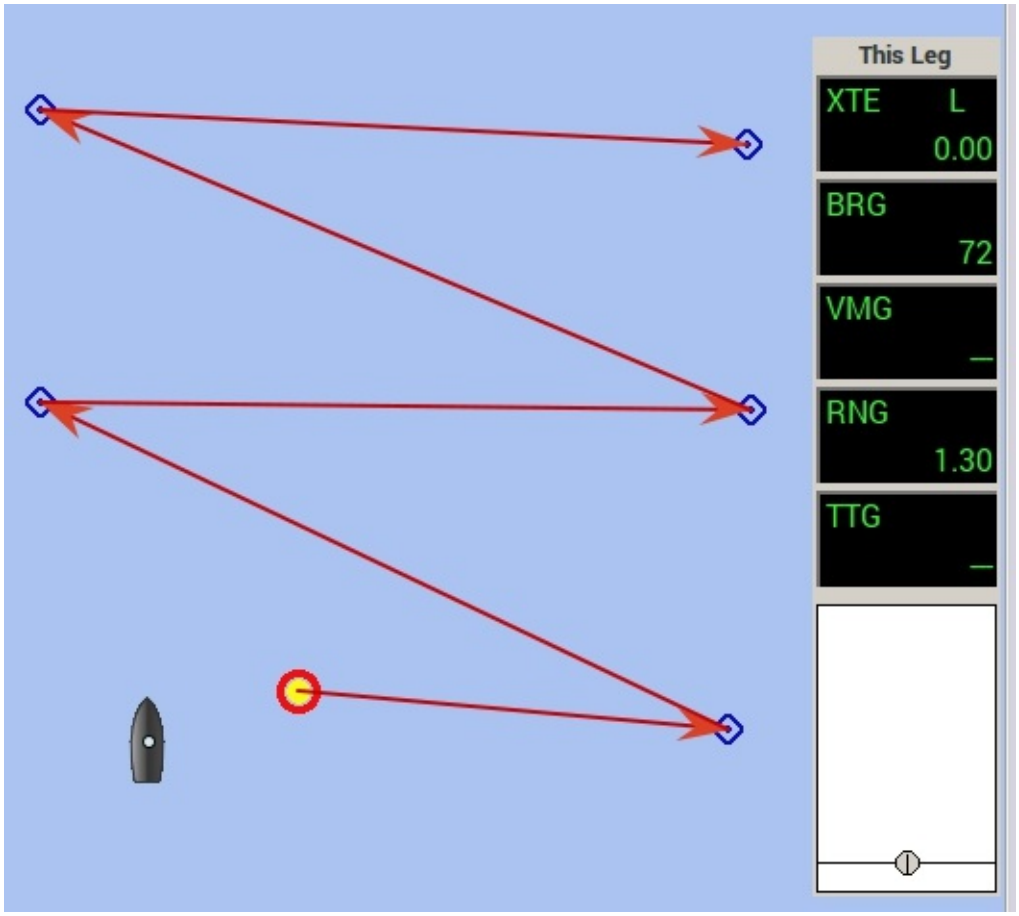


This are the standard connection settings of opencpn to connect to OpenPlotter.



You can activate the route with right mouse button.


Route
Properties...
Activate
Insert Waypoint
Append Waypoint
Copy as KML...
Delete...
Reverse...
Send to GPS



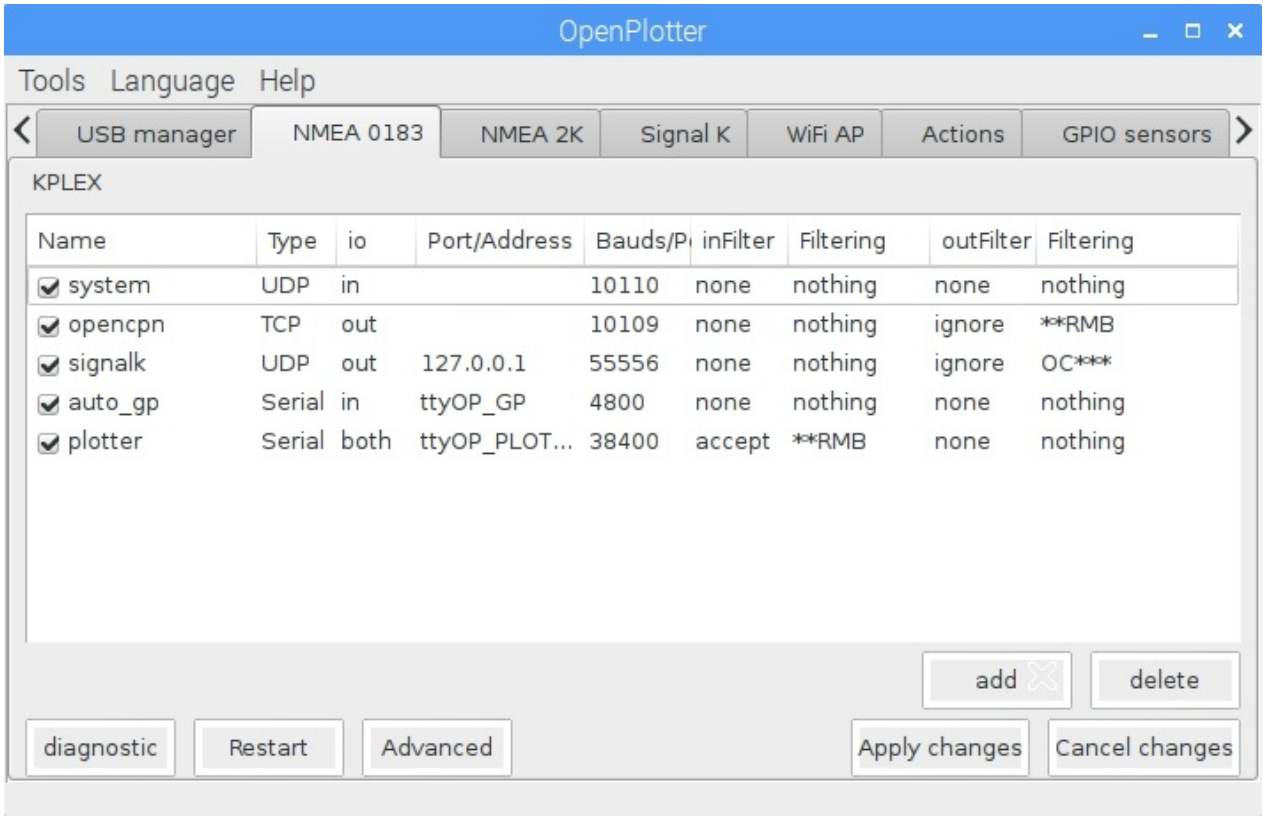
You see a yellow red point blinking. Opencpn sends an NMEA0183 RMB sentence.

If the autopilot has been setup correctly. You can start autopilot to navigate to next waypoint.

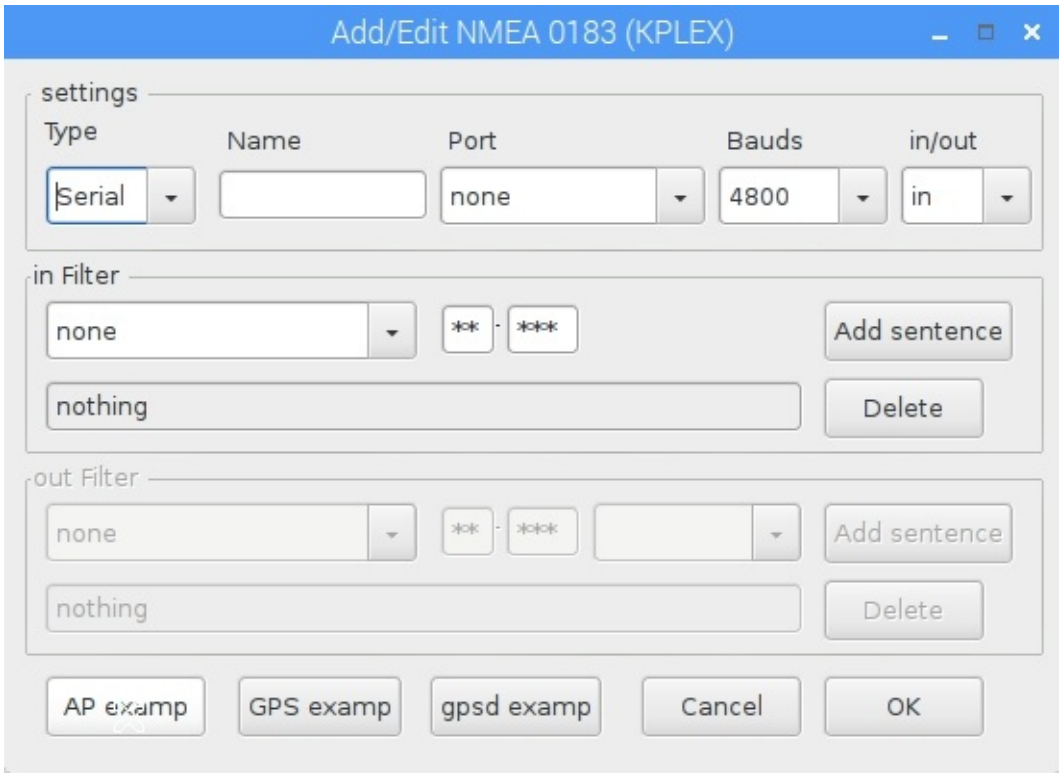
Go on with Sending data to autopilot.

Sending data to the autopilot

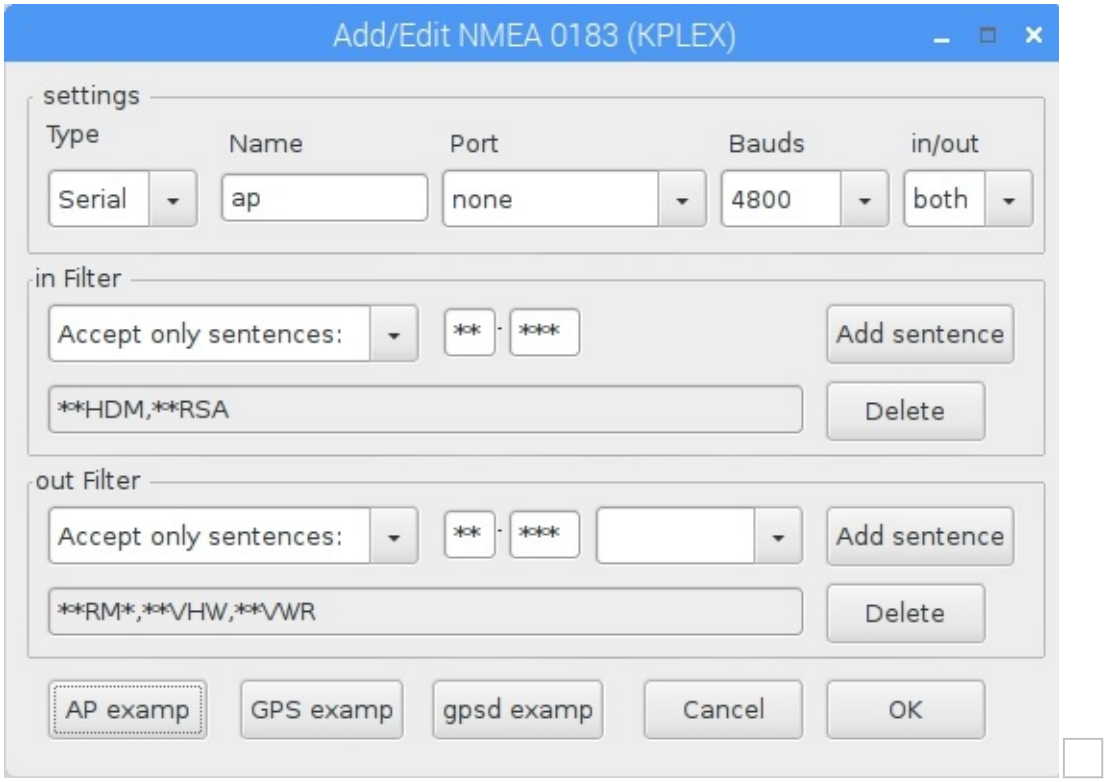
The autopilot has to be setup



Click on **add** (or double click on auto_ap).



Click on **AP examp** to load default settings for autopilot.



Select port ttyOP_AP

in Filter: What sentences are allowed to be received from the autopilot to OpenPlotter/kplex.

Most autopilots have there own fluxgate compass and rudder angle sensor. These data can be used from OpenPlotter.

More important is the

out Filter: What sentences are allowed to be send from OpenPlotter/kplex to the autopilot.

It is important to filter the sentences to reduce the transfer volume. NMEA0183 autopilots work normally with 4800 bauds. There's only space for a few sentences.

Which sentences are important?

- RMB (way to selected waypoint send from a chartplotter)
- RMC (minimum GPS data)
- VHW (waterspeed and heading)
- VWR (wind)
- sometimes APA and APB

Sensors Wiring

Some sensors can be connected to the GPIO of the raspberry pi. If these sensors haven't got a connection to the ground of the boat they are safe. When they are grounded you can get ground loops which could damage the raspberry pi or other things. In these situation it is better to use other short MCU boards like Arduino and isolate them with an USB to USB isolation for example.

I2C

- IMU
- Pressure
- Temperature
- Humidity

1W

- DS18B20

SPI

under construction

Pulse

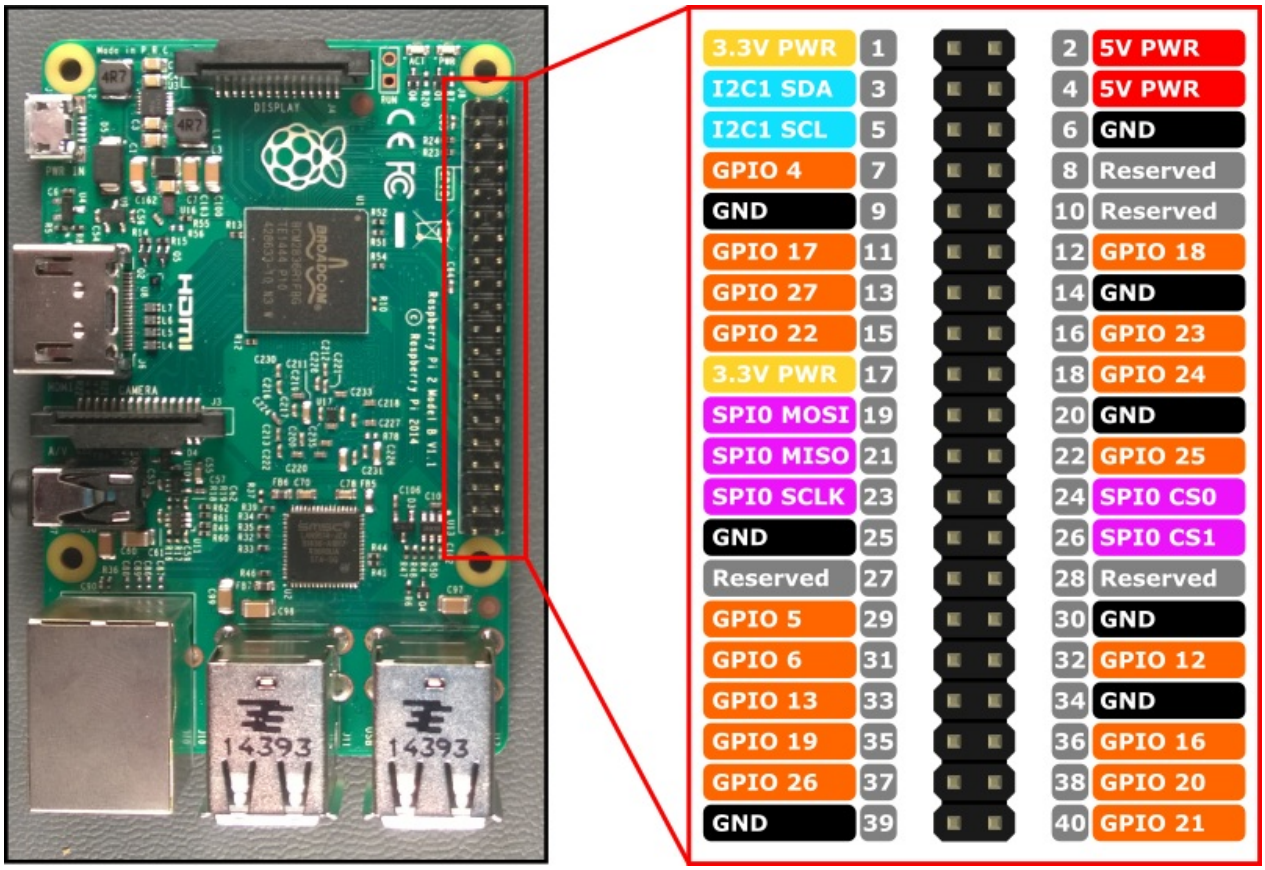
Coming soon

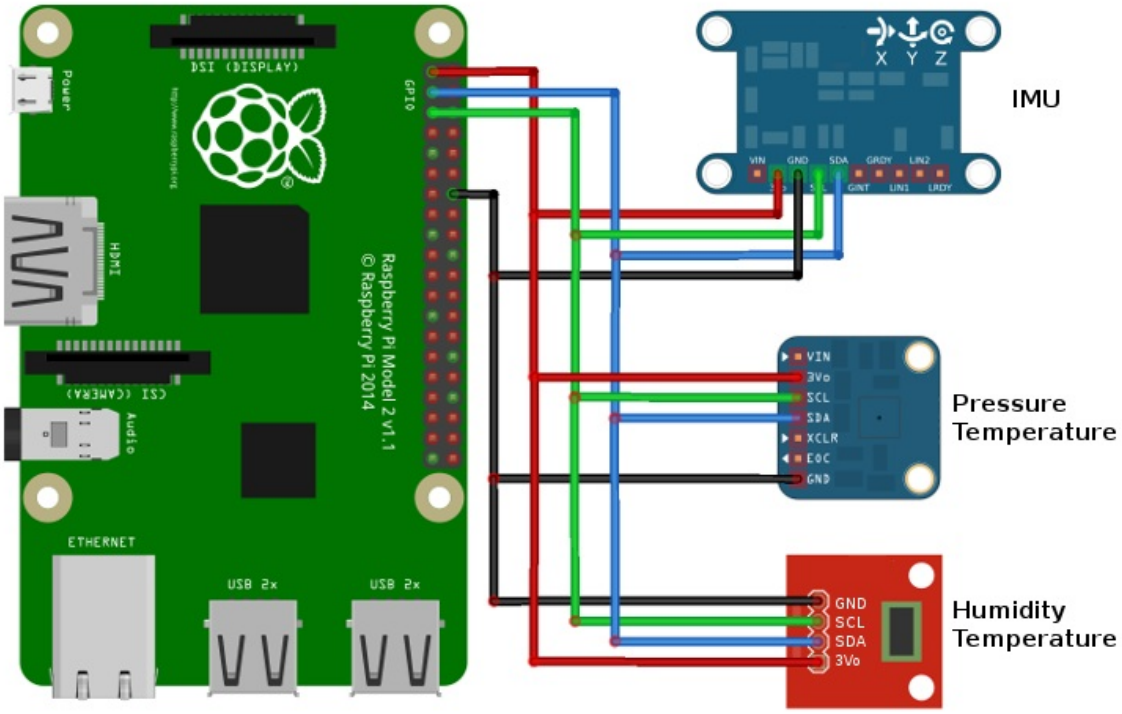
IMU sensor

This chapter is under construction

Wiring

Pins names are according to the diagram below.



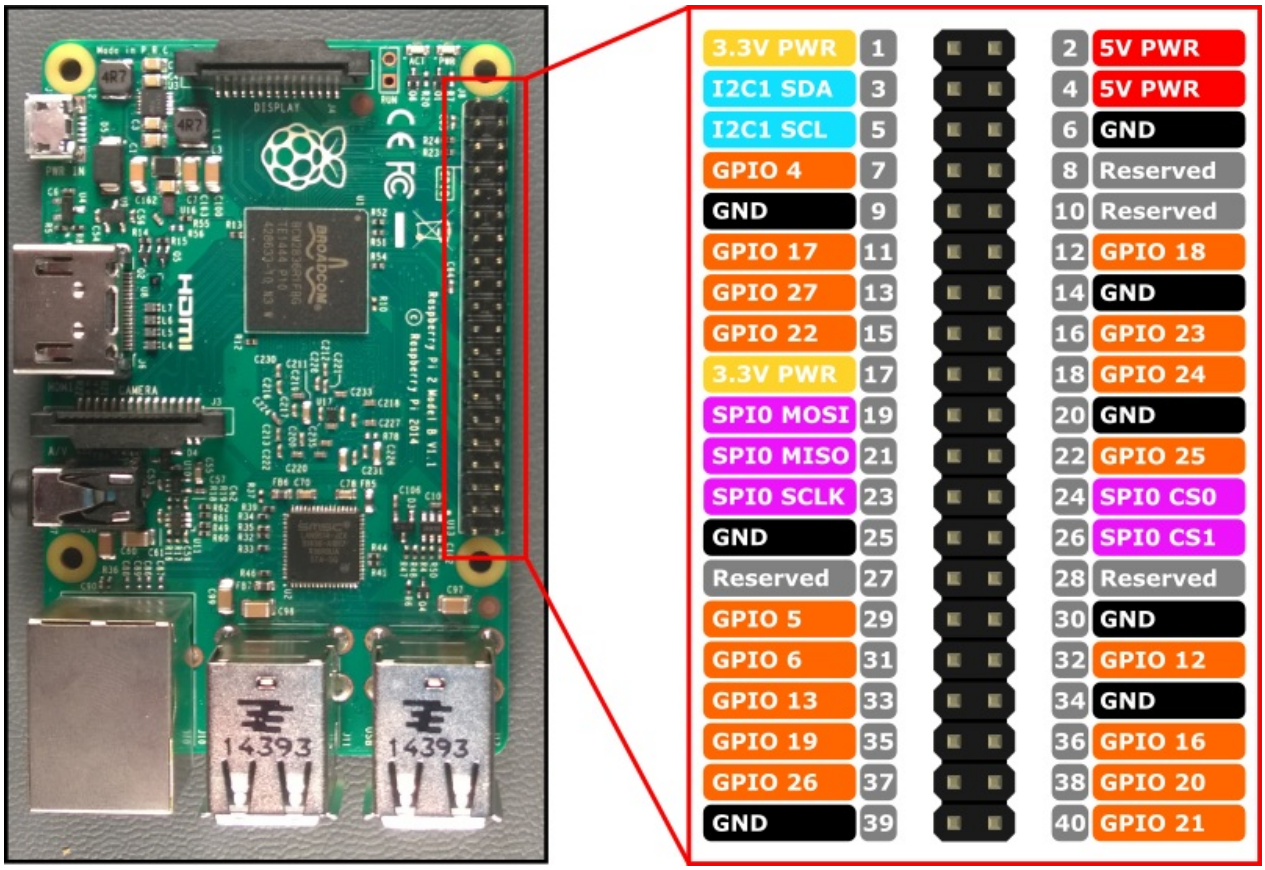

DSI (DISPLAY)
Power
GPIO
HDMI
CSI (CAMERA)
Audio
ETHERNET
USB 2x
USB 2x
Raspberry Pi Model 2 v1.1
© Raspberry Pi 2014
X Y Z
IMU
VIN
GND
SDA
GRDY
LIN2
GINT
LIN1
LRDY
VIN
3Vo
SCL
SDA
XCLR
EOC
GND
Pressure
Temperature
GND
SCL
SDA
3Vo
Humidity
Temperature

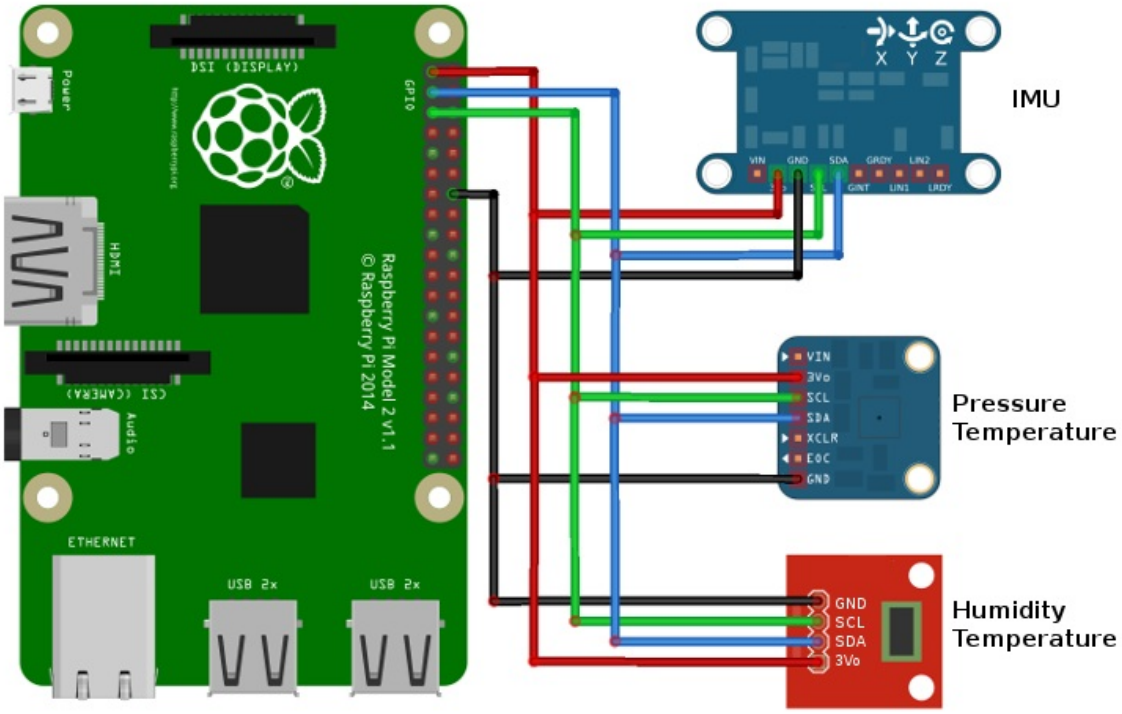
Pressure sensor

This chapter is under construction

Wiring

Pins names are according to the diagram below.



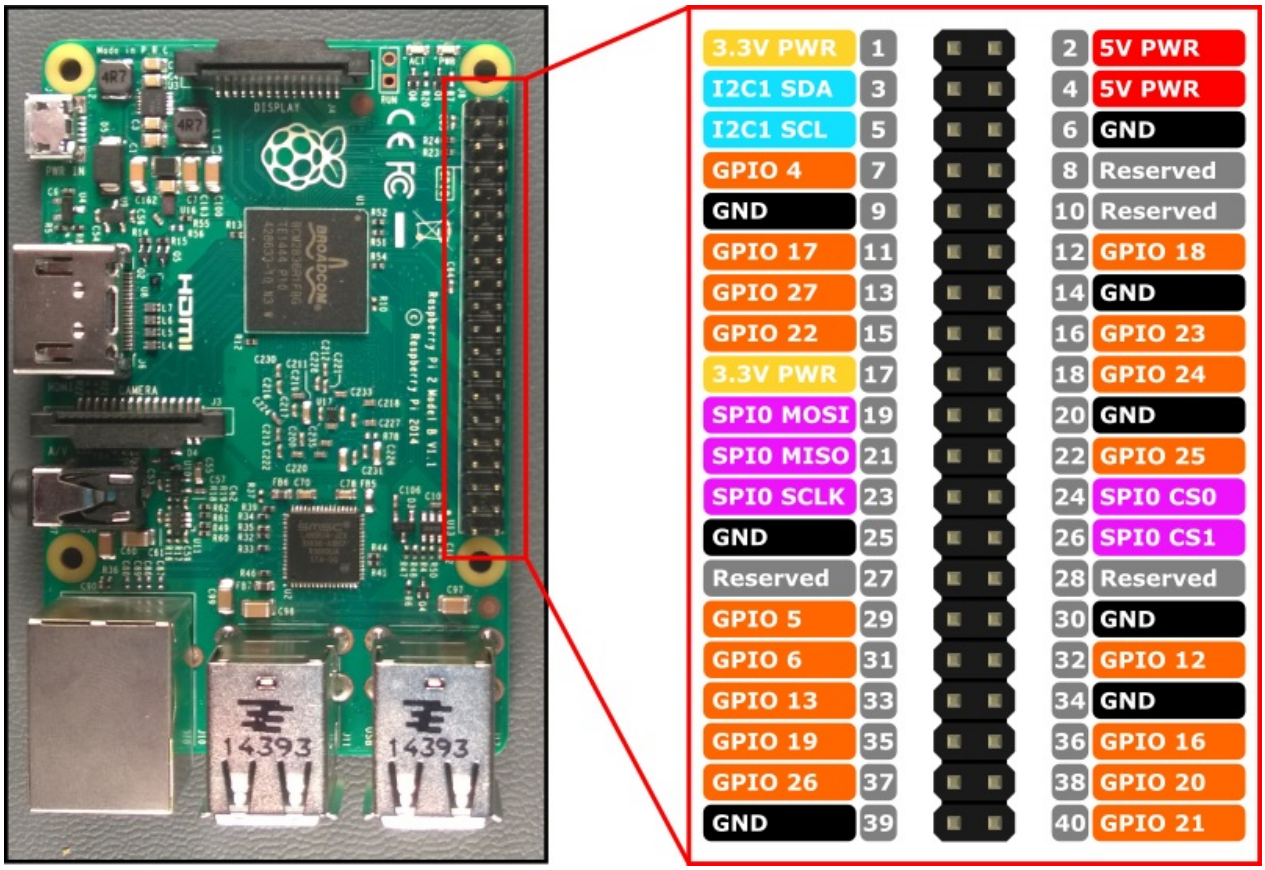

DSI (DISPLAY)
Power
GPIO
HDMI
(CAMERA) CSI
Audio
ETHERNET
USB 2x
USB 2x
Raspberry Pi Model 2 v1.1
© Raspberry Pi 2014
X Y Z
IMU
VIN
GND
SDA
GRDY
LIN2
GINT
LIN1
LRDY
VIN
3Vo
SCL
SDA
XCLR
EOC
GND
Pressure
Temperature
GND
SCL
SDA
3Vo
Humidity
Temperature

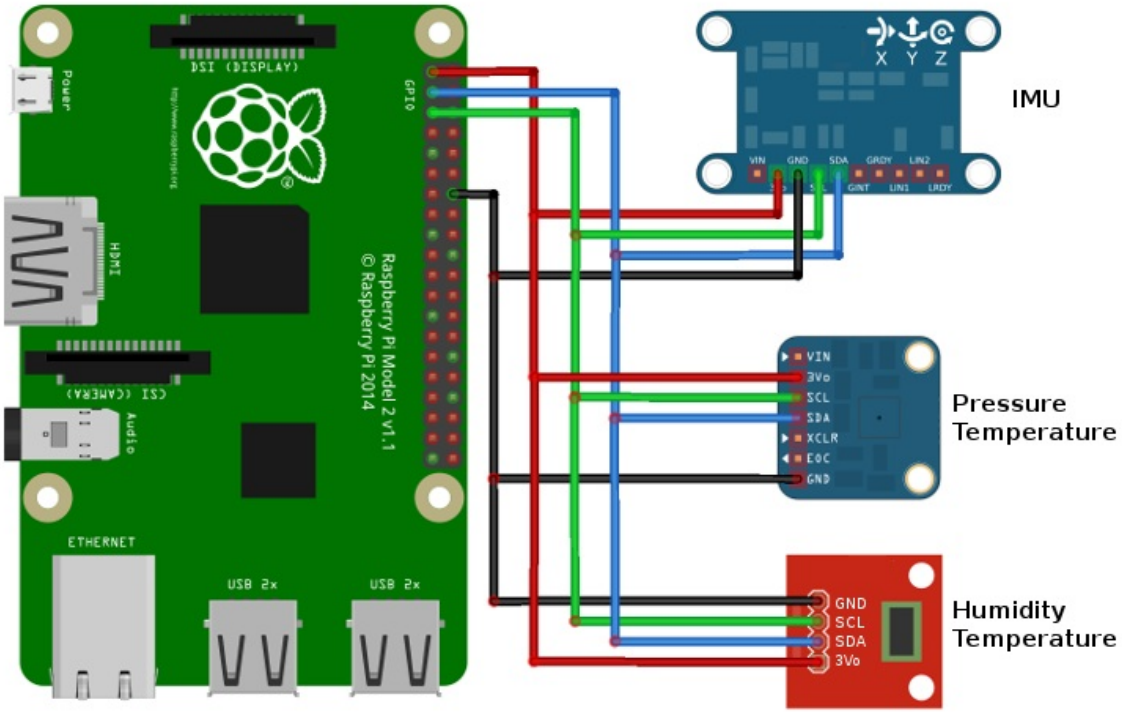
Temperature sensor

This chapter is under construction

Wiring

Pins names are according to the diagram below.



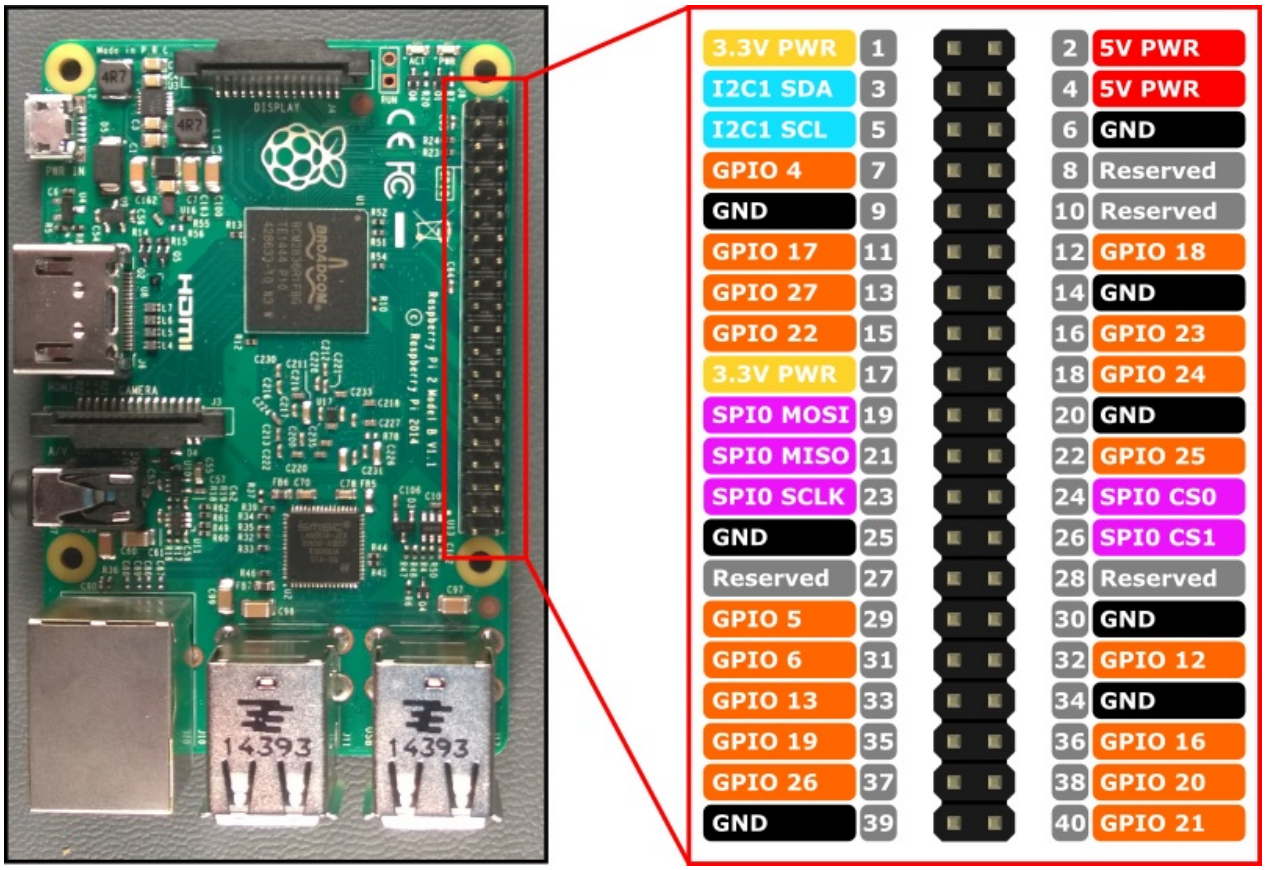

DSI (DISPLAY)
Power
GPIO
HDMI
CSI (CAMERA)
Audio
ETHERNET
USB 2x
USB 2x
Raspberry Pi Model 2 v1.1
© Raspberry Pi 2014
X Y Z
IMU
VIN
GND
SDA
GRDY
LIN2
GINT
LIN1
LRDY
VIN
3Vo
SCL
SDA
XCLR
EOC
GND
Pressure
Temperature
GND
SCL
SDA
3Vo
Humidity
Temperature

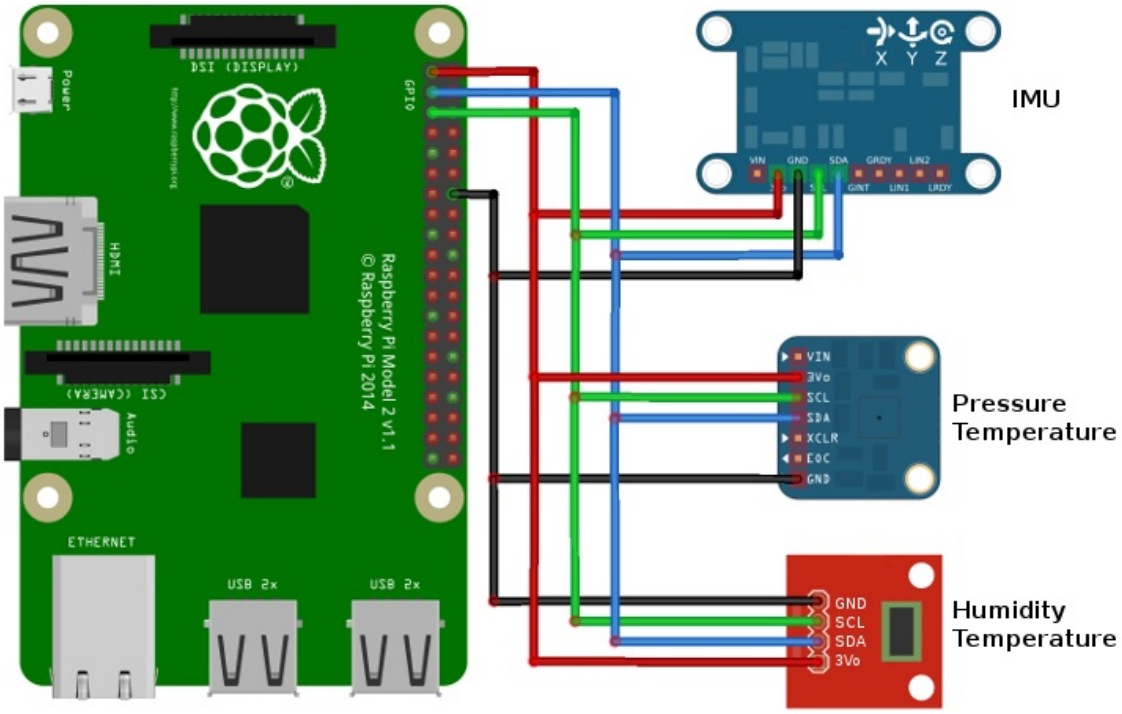
Humidity sensor

This chapter is under construction

Wiring

Pins names are according to the diagram below.



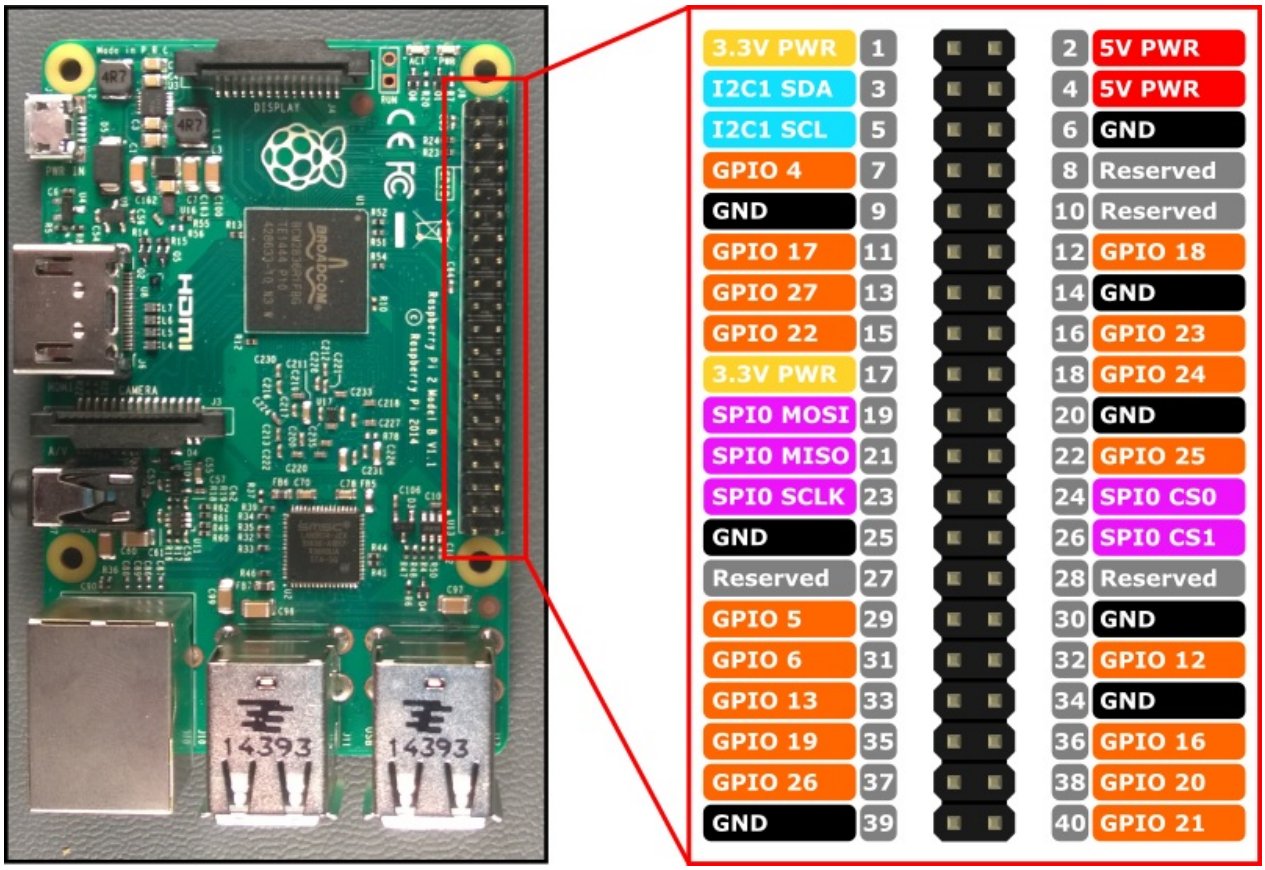

DSI (DISPLAY)
Power
GPIO
HDMI
CSI (CAMERA)
Audio
ETHERNET
USB 2x
USB 2x
Raspberry Pi Model 2 v1.1
© Raspberry Pi 2014
X Y Z
IMU
VIN
GND
SDA
GRDY
LIN2
GINT
LIN1
LRDY
VIN
3Vo
SCL
SDA
XCLR
EOC
GND
Pressure
Temperature
GND
SCL
SDA
3Vo
Humidity
Temperature

DS18B20 temperature sensors

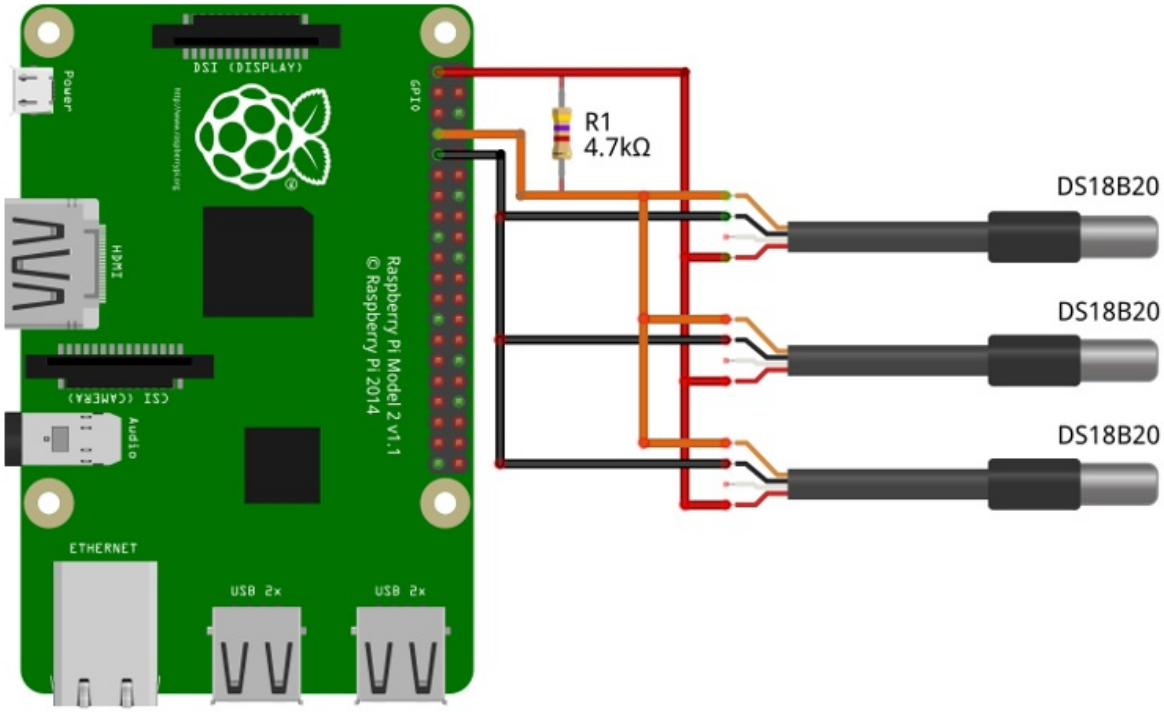
This chapter is under construction

Wiring

Pins names are according to the diagram below.



You have to connect these sensors to **GPIO4**, **GND** and **3.3V** pins. Some sensors may have a fourth wire which do not need to be connected. You need to use a pull-up resistor as shown in the image below. You can connect multiple sensors in paralel using just one resistor.



If you want to change the GPIO pin, edit the file config.txt typing in a terminal:

```
sudo nano /boot/config.txt
```

At the end of the file you should see a line like this

dtoverlay=w1-gpio

replace it by

dtoverlay=w1-gpio,gpiopin=x

where x is your desired GPIO pin. Save and reset.

Settings

Wiring Switches

This chapter is under construction

Switches are driven through the GPIO (General Purpose Input/Output) pins. These pins are copper connected to the CPU chip. A short circuit with 0 V or +3.3 V is not final, but any direct contact with the + 5 V (or higher) can KILL the Raspberry Pi... Switches have to be activated in OpenPlotter related tab.

Common switches

Door switches

Float switches

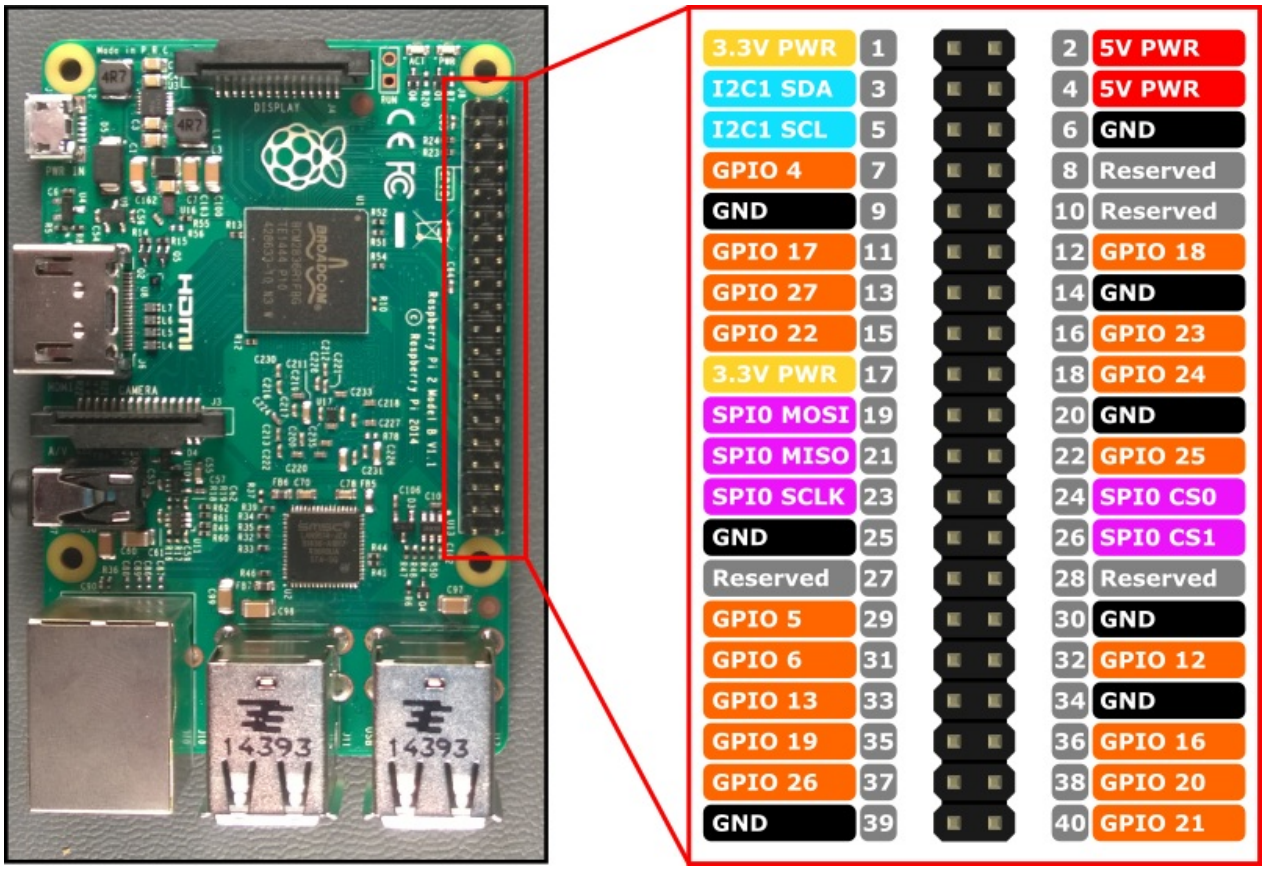
Motion sensor switches

Common switches

This chapter is under construction

Wiring

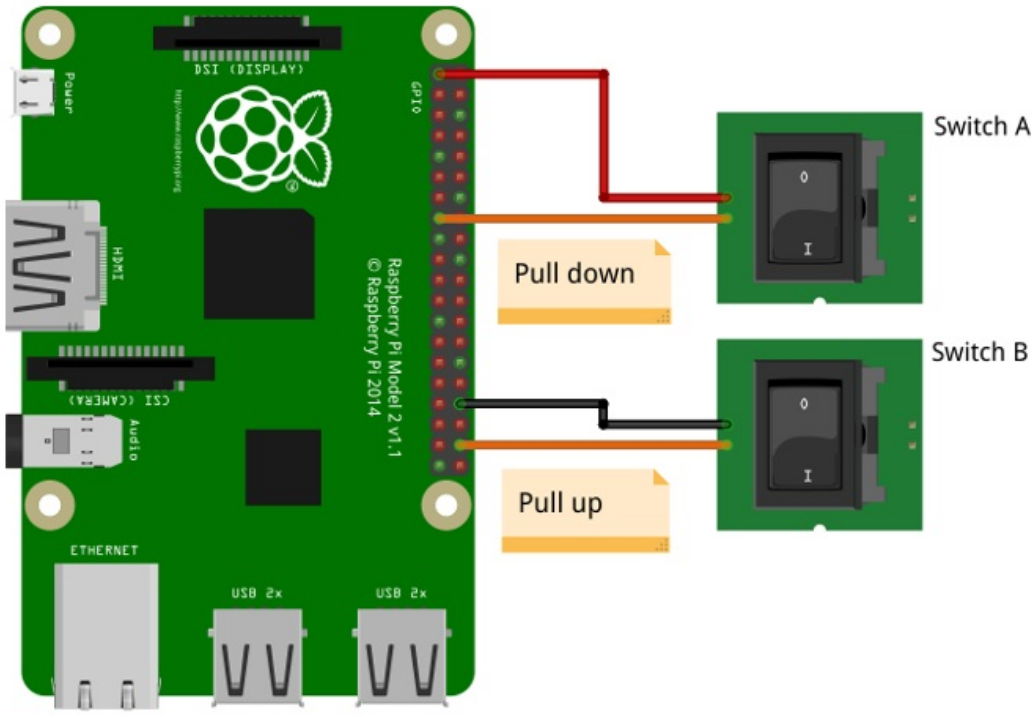
Pins names are according to the diagram below.



For normal switches (open by default), you have to select "Pull down" in "Switches" tab an connect switch between selected GPIO pin and +3.3v pin (DANGER, NEVER TO +5v).

For special switches (closed by default), you have to select "Pull up" in "Switches" tab an connect switch between selected GPIO pin and GND pin.

It is not a problem if you make a mistake connecting to GND or +3.3v but be careful and avoid the +5v pin.


DSI (DISPLAY)
Power
GPIO
HDMI
CSI (CAMERA)
Audio
ETHERNET
USB 2x
USB 2x
Raspberry Pi Model 2 v1.1
© Raspberry Pi 2014
Switch A
Switch B
Pull down
Pull up
0
I
0
I

Settings

Wiring Outputs

This chapter is under construction

Outputs are driven through the GPIO (General Purpose Input/Output) pins. These pins are copper connected to the CPU chip. A short circuit with 0 V or +3.3 V is not final, but any direct contact with the + 5 V (or higher) can KILL the Raspberry Pi...
The output have to be activated in OpenPlotter related tab.

LEDs

Relays

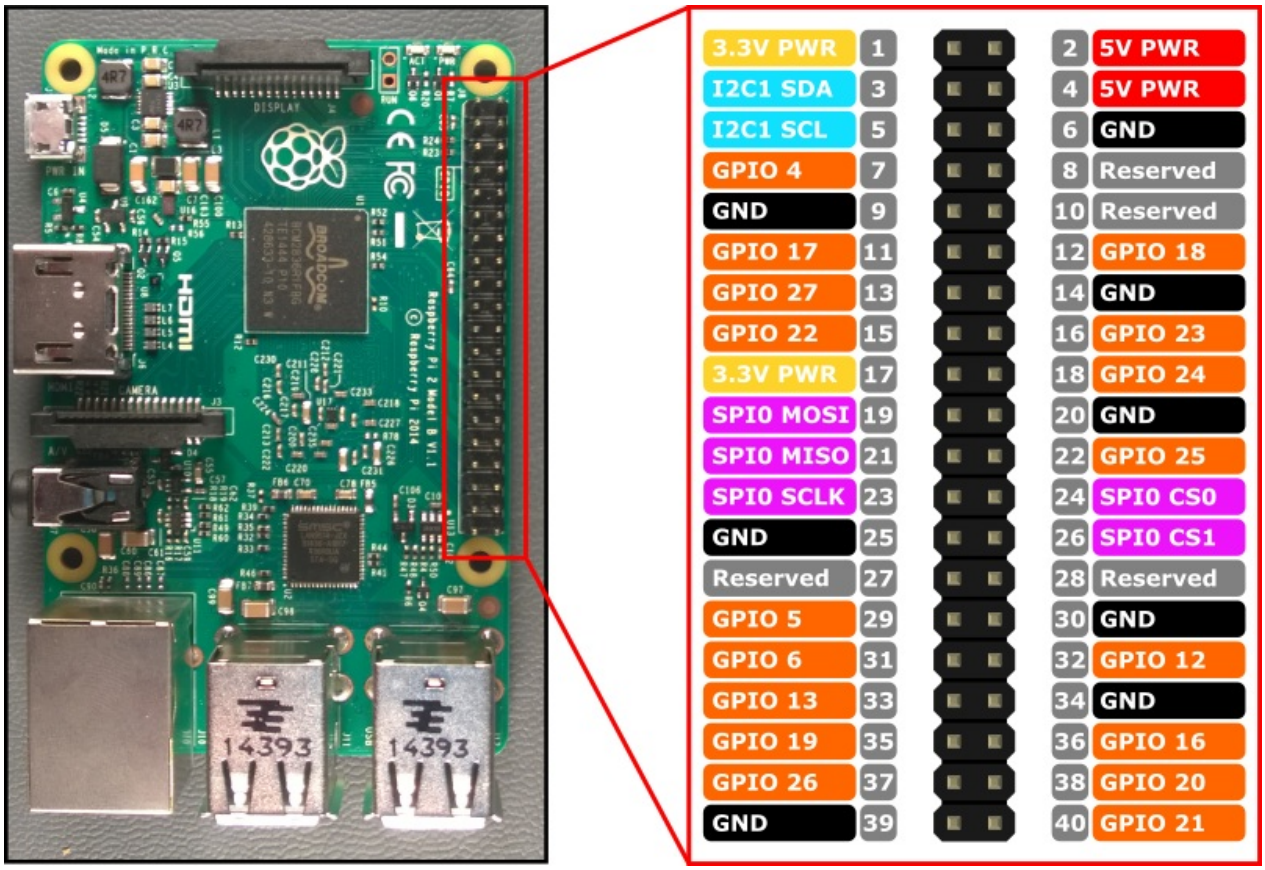
Buzzers

LEDs

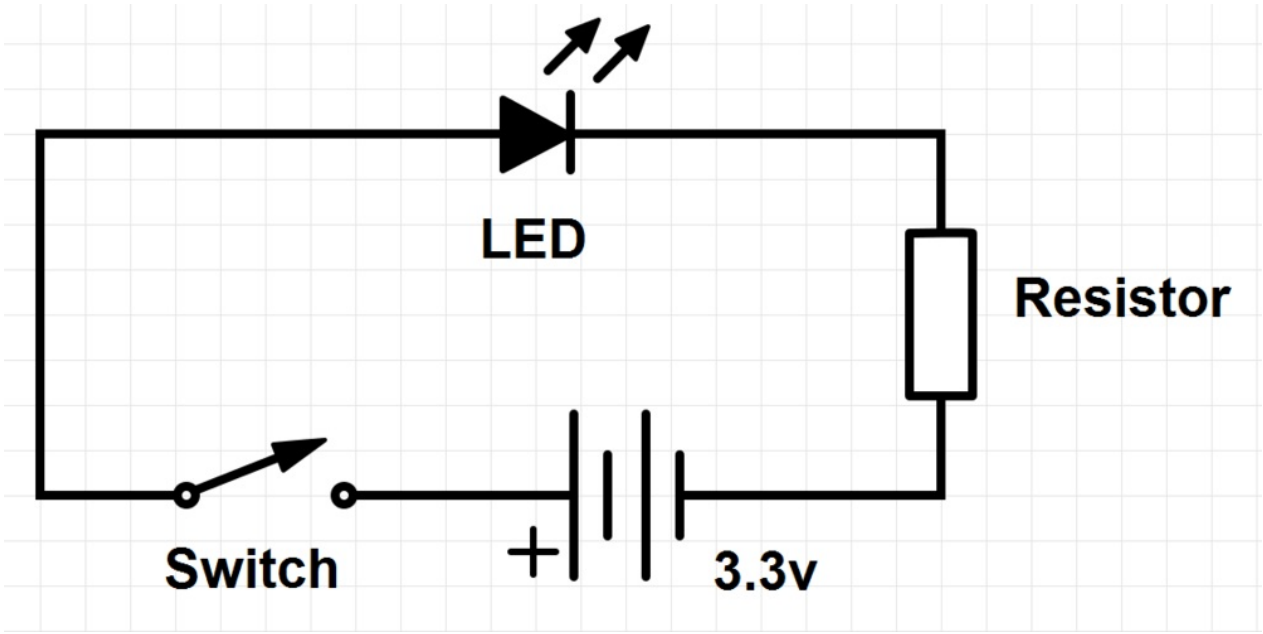
This chapter is under construction

Wiring

Pins names are according to the diagram below.

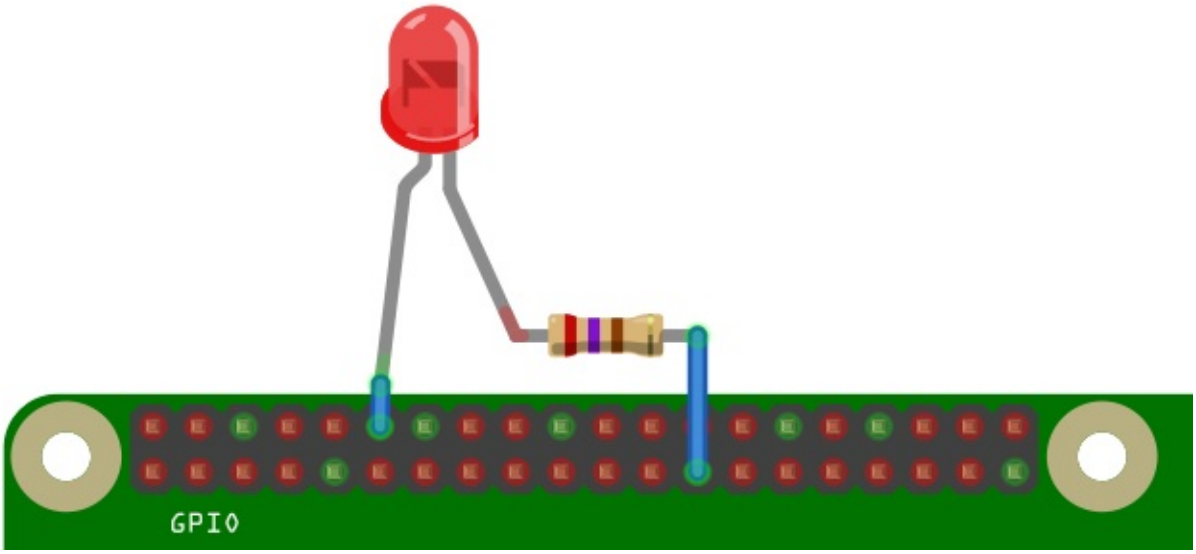


Ignoring the Pi for a moment, one of the simplest electrical circuits that you can build is a battery connected to a light source and a switch (the resistor is there to (limit the current flow) and protect the LED):



When we use a GPIO pin as an output, the Raspberry Pi replaces both the switch and the battery in the above diagram. Each pin can turn on or off,or go **HIGH** or **LOW** in computing terms. When the pin is **HIGH** it outputs **3.3 volts** (3v3); when the pin is LOW it is off.

Here's the same circuit using the Raspberry Pi. The LED is connected to a GPIO pin (which can output +3v3) and a ground pin which is 0v and acts like the negative terminal of the battery : (the resistor is there to limit the current flow and protect both the LED and the Raspberry CPU). A value of 1 kOhm will retrain the current around 1.5 mA



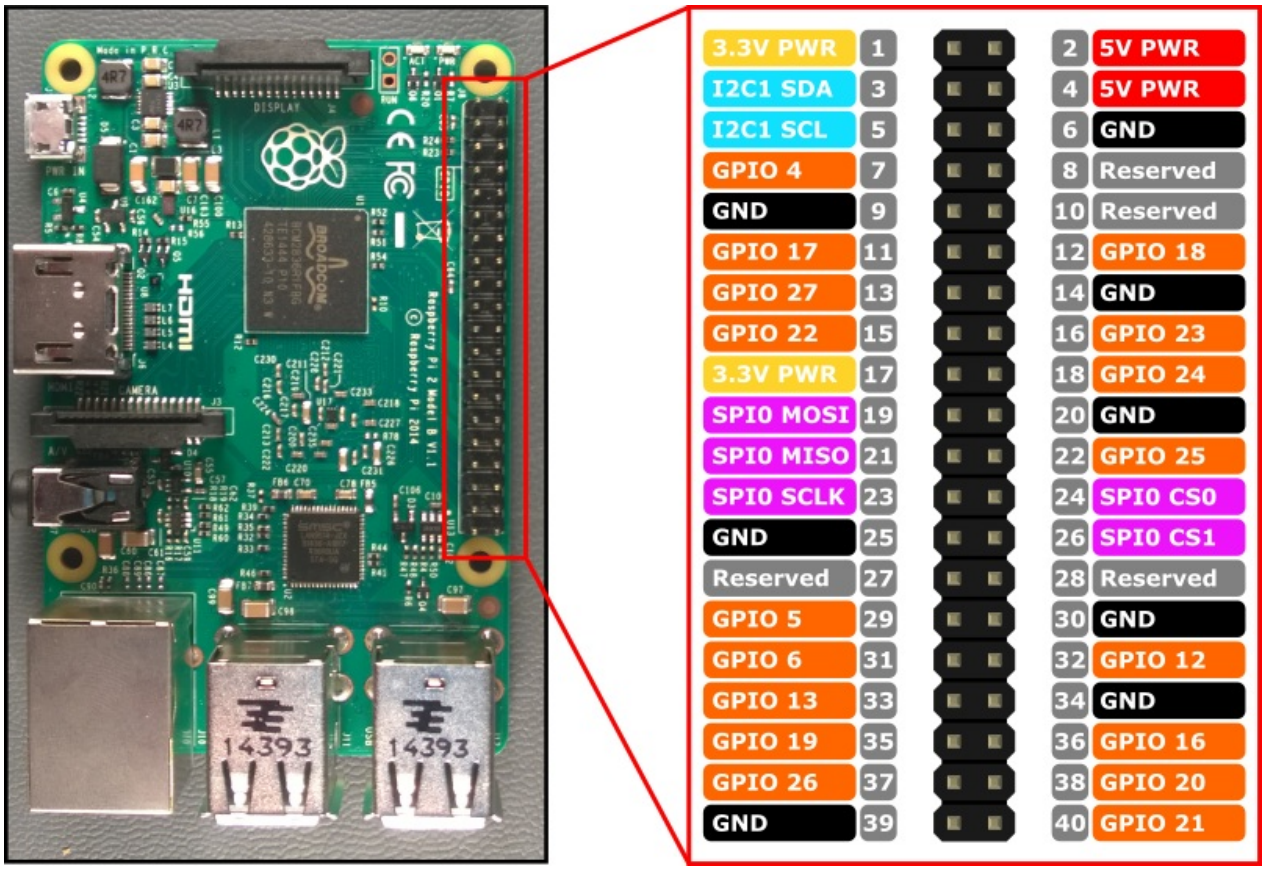
Settings

Relays

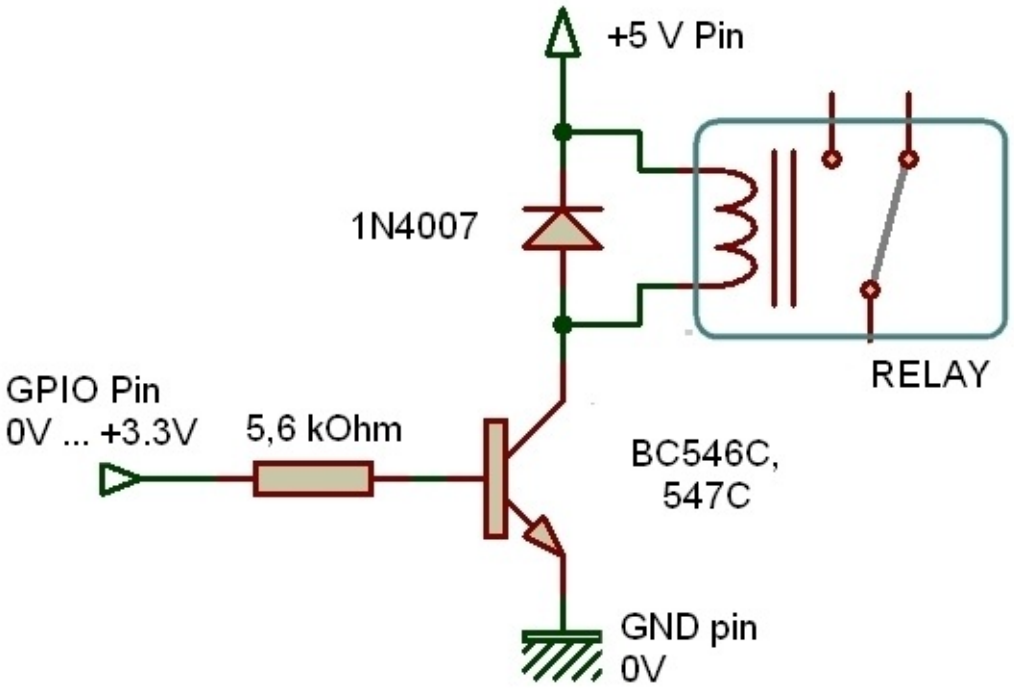
This chapter is under construction

Wiring

Pins names are according to the diagram below.



The GPIO pins can only deliver a few mA, and cannot drive directly a relay. Therefore a few electronics components are required :



- The NPN transistor BC54xC acts as a valve to control the current through the relay coil (switching mode). It is either passing when GPIO is HIGH, or blocked when GPIO is LOW. Its maximum collector current is 100 mA. The final "C" ensures than its high gain (>400) will minimize the current drawn from the GPIO.
- The 5,6 kOhm resistor limits the current drawn from the GPIO pin below 0,5 mA ;
- The diode, 1N4007 prevents damaging the transistor (and/or GPIO pins !) when the relay is commuted.
- The relay is a 5VDC relay, with an internal resistance of 60 Ohm. It requires less than 90mA, compatible with the transistor. (e.g. printed circuit relay Omron G5RL-1-E-HR 5 VDC 5 V/DC 16 A). It is available from Conrad on line shops.

Relays used in automobile industry should be connected to + 12 VDC. They will draw a higher current (4 to 500 mA), and call for a more elaborated design (cascading two transistors).

Settings

License

OpenPlotter license

OpenPlotter is free software: you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 2 of the License, or any later version.

OpenPlotter is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with OpenPlotter. If not, see http://www.gnu.org/licenses/.

This manual license

This manual is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

http://creativecommons.org/licenses/by-sa/4.0/