1. Funktion

Der Sensor erfasst Flüssigkeitsstände in einem Tank und gibt einen entsprechenden NMEA-Datenstrom per WLAN an das Bordnetzwerk. Dieser kann dann in z.B. OpenCPN visualisiert werden.

Aufgrund der Eigenschaften des Sensors muss kein Loch in den Tank gebohrt werden, daher bietet sich dieser Sensor als Nachrüstlösung bei bestehenden Tanks ohne Höhenstandsanzeige an.
Durch die WLAN-Anbindung müssen keine weiteren Datenkabel verlegt werde, eine 12V Versorgung in der Nähe des Tanks, an die der Mikrocontroller angeschlossen werden kann, ist ausreichend.

Das Modul Wemos D1 mini pro ist ein Mikrocontroller auf Basis des ESP 8266 mit eingebautem WLAN-Modul.
An dem Mikrocontroller wird ein Ultraschallsensor DS1603L angeschlossen.
Der Ultraschallsensor wird außen an den Tankboden angeklebt.
Damit kann dann der Flüssigkeitsspiegel im Tank erfasst werden.
Achtung: Der Sensor muss am Tankboden angebracht werden, er muss also von unten nach oben „pingen“ um den Flüssigkeitsspiegel zu erfassen. Von oben nach unten „pingen“ funktioniert nicht.

Über das WLAN-Modul wird der Wemos mit dem Bordnetzwerk verbunden.
Dazu wird bei erstmaligem Start bzw. jedes Mal, wenn kein bekanntes WLAN-Netz vorhanden ist eine Einrichtungsseite aufgerufen.

Ist der Mikrocontroller mit dem Netzwerk verbunden und hat der Sensor einen Flüssigkeitsspiegel im Tank erkannt werden entsprechende NMEA-Datensätze in das Bordnetzwerk gesendet.

2. Anschluss Sensor an den Wemos

Ich empfehle passend zu dem Wemos D1 mini pro ein Stromversorgungsshield zu besorgen.
Das Wemos Board bekommt man z.B. hier:
https://www.makershop.de/plattformen/d1-mini/d1-mini-pro/

Das passende Stromversorgungsshield wäre dann so etwas:
https://www.makershop.de/module/step-downup/wemos-d1-power-shield/

Mit dem Shopbetreiber habe ich keinerlei Verbindung, ich bin nicht Verwandt oder verschwägert.

Der Sensor hat vier Leitungen, rot, schwarz, gelb und weiß.

Rot	-> +5V
Schwarz	-> GND
gelb	-> D4
weiß	-> D3

Die beiden Anschlüsse D3, D4 werden im Sketch in Zeile 62 und 63 definiert.
Bei Nutzung anderer Eingänge muss das dort angepasst werden.

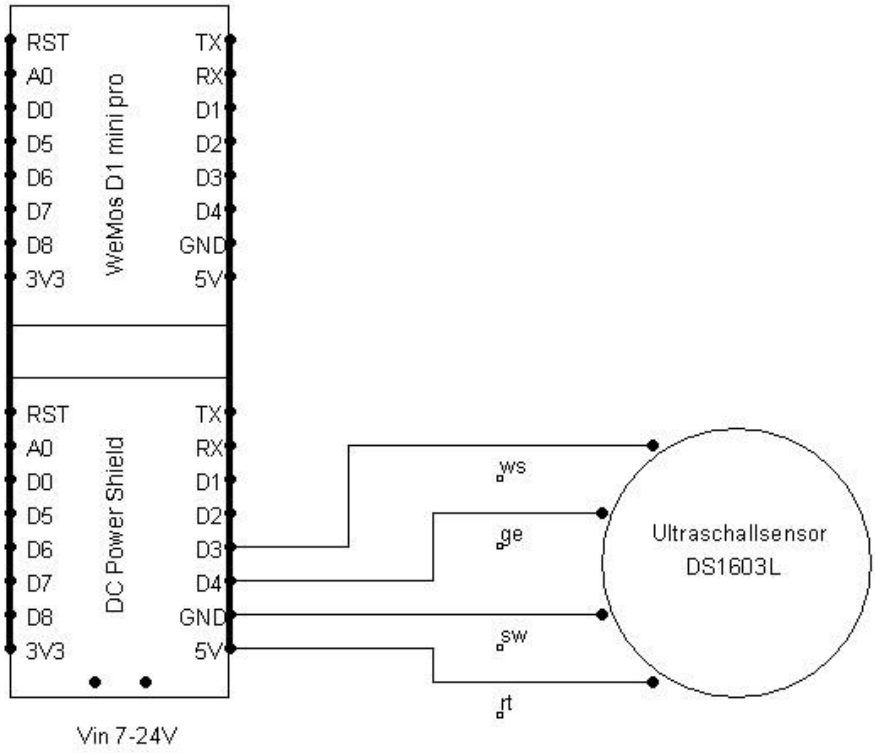

Abbildung 1 Prinzipschaltbild. Der Wemos und das Shield werden zusammengesteckt (s. Kapitel Bilder)

3. Übertragen des Programms an den Wemos

Der Wemos wird über die Arduino IDE (Programmierumgebung) geflasht.
Dazu muss neben der IDE in aktueller Fassung auch zusätzlich ein passender USB-Treiber installiert werden.
Nähere Anleitungen dazu findet man z.B. hier: https://arduino-projekte.info/wemos-d1-mini-pro/

In der IDE wählt man als Board dann LOLIN (Wemos) D1 mini pro.

Es müssen einige Bibliotheken über die Bibliotheksverwaltung in der IDE geladen werden, sofern nicht schon vorhanden:

ESP8266WiFi.h	//https://github.com/esp8266/Arduino
DNSServer.h	Standard Arduino IDE
ESP8266WebServer.h	Standard Arduino IDE
WiFiManager.h	//https://github.com/tzapu/WiFiManager
WiFiUdp.h	Standard Arduino IDE
SoftwareSerial.h	Standard Arduino IDE
MovingAverage.h	Standard Arduino IDE

DS1603L.h
//https:///https://github.com/tzapu/WiFiManager/github.com/wvmarle/Arduino_DS1603L

4. WLAN Einrichtung

Das Programm schaltet bei erstmaligem Gebrauch bzw. jedes Mal wenn keines der bekannten WLAN-Netzwerke gefunden wird ein eigenes WLAN ein, in dem eine Einrichtungsseite aufrufbar ist.

Dieses WLAN muss im Sketch entsprechend eingerichtet werden.
Es ist ein Name (SSID) und ein Password (mindestens 7 Zeichen lang) zu vergeben.
Dieses muss an zwei Stellen im Sketch erfolgen:
In Zeile 104 sowie in Zeile 134 bitte die entsprechenden Eintragungen machen.

Die Einrichtungsseite ist dann z.B. per Handy aufrufbar in dem man sich mit dem WLAN des Wemos verbindet und mit dem Handy im Internetbrowser eine neue Seite aufruft.

Anschließend kann man dort die nötigen Eintragungen machen um den Wemos mit einem Netzwerk zu verbinden.

Die Einrichtungsseite / das WLAN ist für 120 Sekunden eingeschaltet. Dann wird das WLAN-Modul des Wemos für ca. 3 Minuten abgeschaltet (Stromsparfunktion) anschließend wird der Wemos neu gestartet und die Einrichtungsseite wieder aufgerufen.
Die Zeiten können im Sketch in Zeile 100 sowie in Zeile 109 angepasst werden.

Geht die Verbindung zum WLAN verloren wird die Einrichtungsseite entsprechend wie zuvor beschrieben aufgerufen.

Erkennt der Wemos beim Start ein bereits bekanntes WLAN oder kehrt die WLAN-Verbindung während des Betriebs des Wemos nach einem vorherigen Verbindungsverlust zurück erfolgt automatisch eine Verbindung mit diesem WLAN ohne dass erneut die Verbindungsdaten eingegeben werden müssen.

Die Einrichtungsseite ist selbsterklärend.

Der Wemos überträgt 4 Pakete (Zeile 160 im Sketch ändern wenn man mehr oder weniger Pakete übertragen möchte). Anschließend wird das WLAN-Modul abgeschaltet (der Wemos arbeitet in dieser Zeit aber weiter) um Strom zu sparen. Nach Ablauf einer vorgegebenen Zeit (Zeile 55 im Sketch) schaltet sich das WLAN Modul wieder ein, der Wemos verbindet sich automatisch mit dem Netz, sendet die festgelegte Anzahl Pakete und schaltet dann das WLAN Modul wieder ab.
Der Stromverbrauch beträgt ohne Abschaltung mit Sensor ca. 90 mA/h und mit Abschaltung ca. 45 mA/h.

5. Einrichtung in OpenCPN

In OpenCPN kann man den NMEA-Datenstrom z.B. mittels des EngineDashboard-Plugin in Form von Rundinstrumenten visualisieren.

Der Datenstrom wird über WLAN an die Broadcastadresse des örtlichen Netzwerkes übertragen.
Diese Adresse ist festgelegt und ist immer die xxx.xxx.xxx.255.
Der Datenport ist 50000 (Festgelegt in Zeile 50).
Das Übertragungsprotokoll ist UDP.

In OpenCPN wird also eine Netzwerkverbindung als Datenverbindung zusätzlich zu bereits eingerichteten Verbindungen definiert (unter Einstellungen):

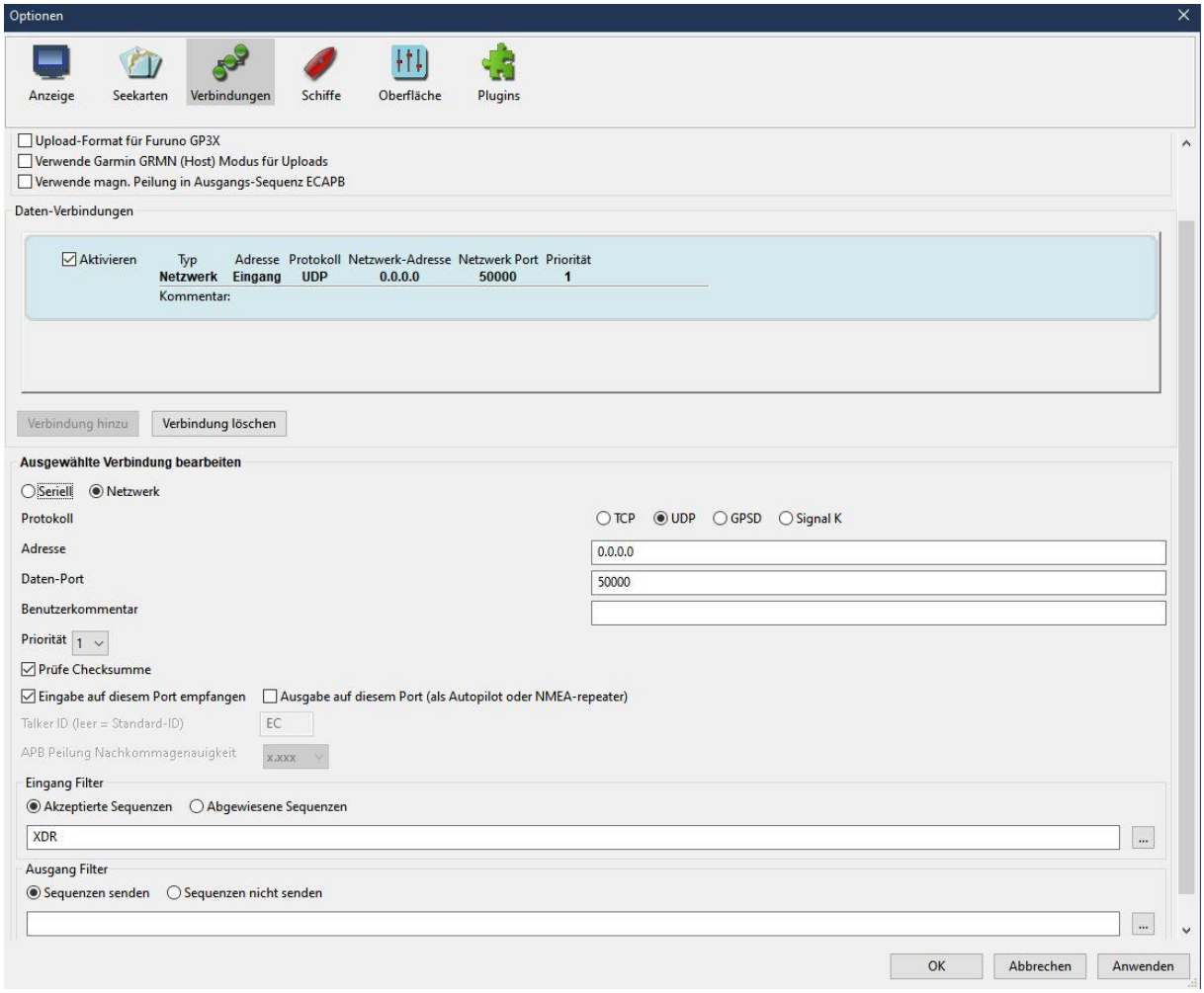

Abbildung 2 Datenverbindung in OpenCPN einrichten

Die Adresse darf 0.0.0.0 sein, da die Broadcastadresse von OpenCPN automatisch abgefragt wird.

Als nächstes muss zur Visualisierung in OpenCPN das EngineDashboard-Plugin heruntergeladen werden.
Quelle: https://github.com/TwoCanPlugIn/EngineDashboard

Freundlicherweise sind unter dem Ordner „prebuilt“ bereits einige fertige Installationspakete vorhanden, so dass man nicht unbedingt selber compilieren muss.

Hat man das Plugin installiert und in OpenCPN aktiviert stellt man folgende Instrumente ein (die Einstellung erfolgt wie bei dem Standart-Dashboard):

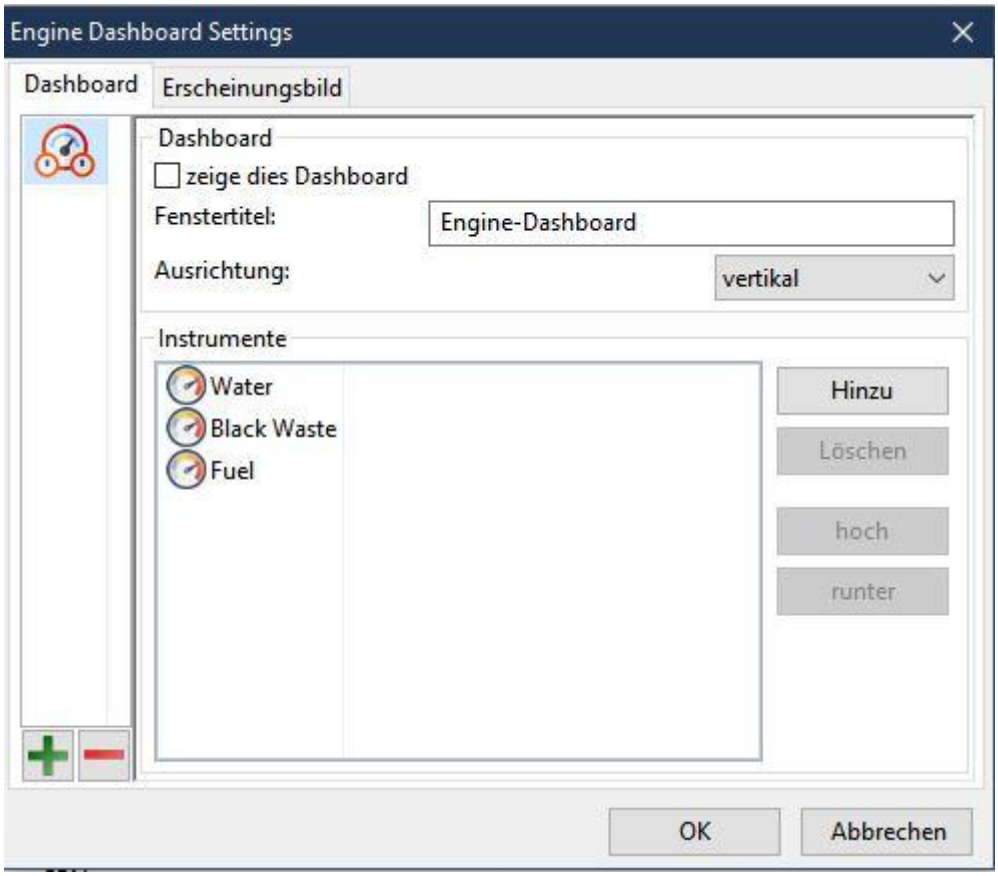

Abbildung 3 Einstellungen EngineDashboard

Welche Instrumente dann genau benutzt werden sollen kann man im Sketch mittels der NMEA-XDR-Sequenz anpassen (Zeile 230). Dazu bitte auch die Dokumentation des Plugins konsultieren.

Das EngineDashboard Plugin überwacht den NMEA-Datenstrom. Werden über einen Zeitraum > ca. 5 sek keine XDR-Sequenzen empfangen fallen die Instrumente alle auf 0.
Da der Wemos aber nur alle 3 Minuten sendet, würden die Instrumente zwischenzeitlich 0 Anzeigen. Um das zu verhindern muss man noch zusätzlich das Plugin „NMEA-Converter“ in OpenCPN aktivieren.
Mit Hilfe dieses Plugins kann man einen „Fake“Datenstrom senden, der dann dazu führt, dass die Instrumente zwischenzeitlich nicht 0 anzeigen.

Man erstellt dazu in den Einstellungen einfach unter „Neu“ eine XDR-Sequenz die man alle 3 sek senden lässt:

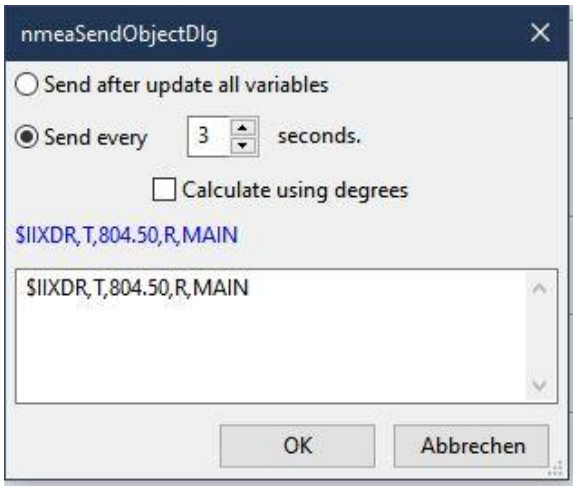

Abbildung 4 Einstellungen "Fake"NMEA Datenstrom

Mit dieser Konfiguration würde man allerdings einen möglichen Ausfall des Sensors bzw. Mikrocontrollers nicht sehen können (die Anzeigen würden zunächst immer weiter den letzten empfangenen Wert anzeigen). Daher wird von dem Wemos zusammen mit der eigentlichen XDR-NMEA-Sequenz für den Tankinhalt auch noch zusätzlich eine weitere NMEA Sequenz mit zwei Kennbuchstaben am Anfang gesendet (im Code AAXDR).
Diese Kennung kann bei mehreren Wemos-Kontrollern angepasst werden (jeder der Kontroller sollte eine eigene Kennung bekommen). Die Anpassung erfolgt in Zeile 163.

Hinweis:
Wenn man an dieser Stelle eine andere Kontrollkennung (z.B. BBXDR) verwendet muss die Checksumme (die beiden Zahlen nach dem *) angepasst werden. Dazu kann man entsprechende online-tools zur Berechnung der Checksumme verwenden, z.B. hier:
https://www.scadacore.com/tools/programming-calculators/online-checksum-calculator/

Man gibt dann dort unter dem ASCII Feld alle Zeichen (einschl. der Kommas) zwischen dem $ und dem * ein und bekommt dann als Ergebnis die Checksumme die nach dem * eingetragen werden muss.

Die jeweilige spezielle NMEA Datenkennung lässt sich in OpenCPN mittels dem Plugin „Watchdog“ überwachen. Hier kann man dann festlegen, dass die entsprechende NMEA-Sequenz z.B. alle 200 sek empfangen werden muss. Ansonsten wird eine Alarmmeldung generiert.

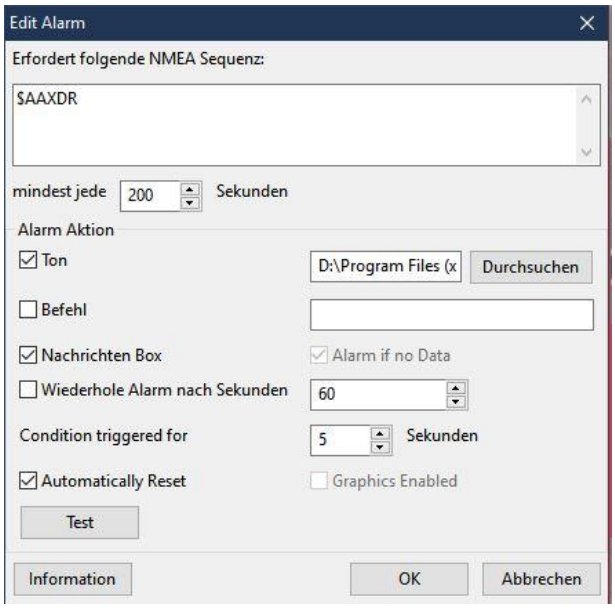

Abbildung 5 Einstellungen Watchdog

6. Auswertung Sensorwerte

Die Sensorwerte werden alle 5 Sekunden in ein 10 stelliges Schieberegister geschrieben (FIFO First in First out) und damit dann eine gleitender Mittelwert gebildet.

In den Zeilen 214 bis 217 erfolgt eine einfache Prozentuale Umrechnung auf der Basis einer festen Tankhöhe.
Die Höhe bzw. die Umrechnung generell muss ggf. an die Tankgeometrie angepasst werden.
Dazu ist dann an dieser Stelle der Algorithmus entsprechend zu Ändern.

7. Bilder

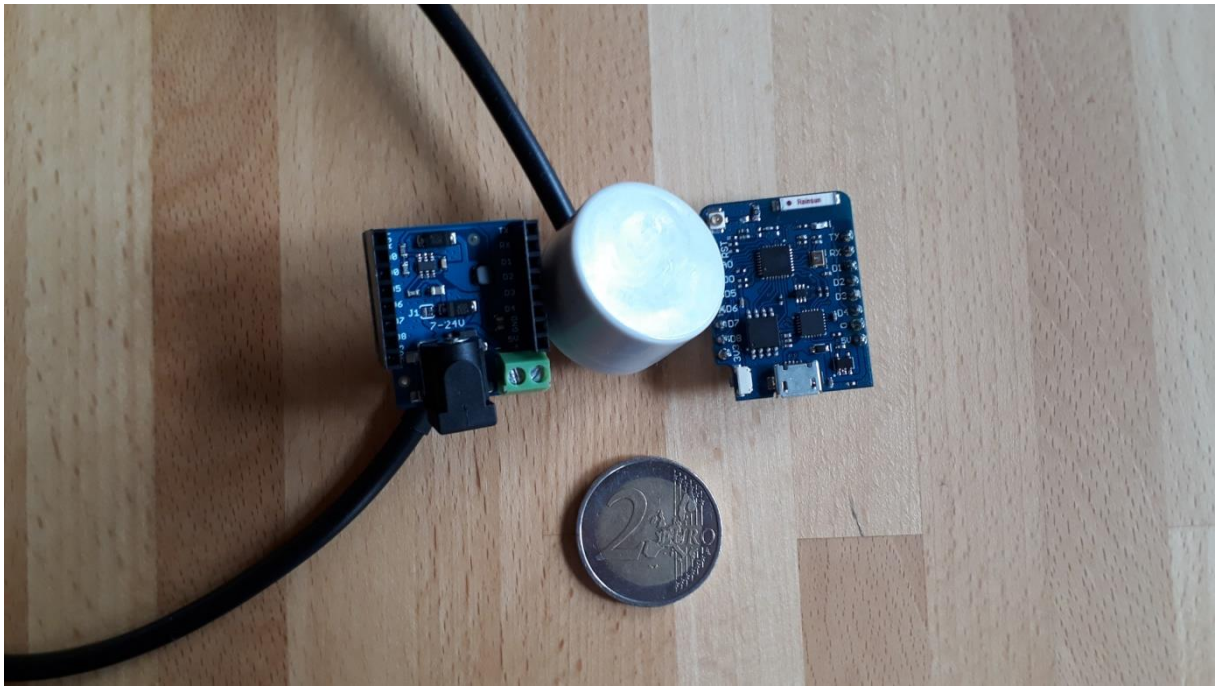

Abbildung 6 Stromversorgungsshield, Sensor, Wemos (v.l.) Zum Größenvergleich 2-Euro Münze

Abbildung 7 Wemos und Shield zusammengesteckt

Abbildung 8 Sensor unter dem "Testtank"

Abbildung 9 Der Testtank

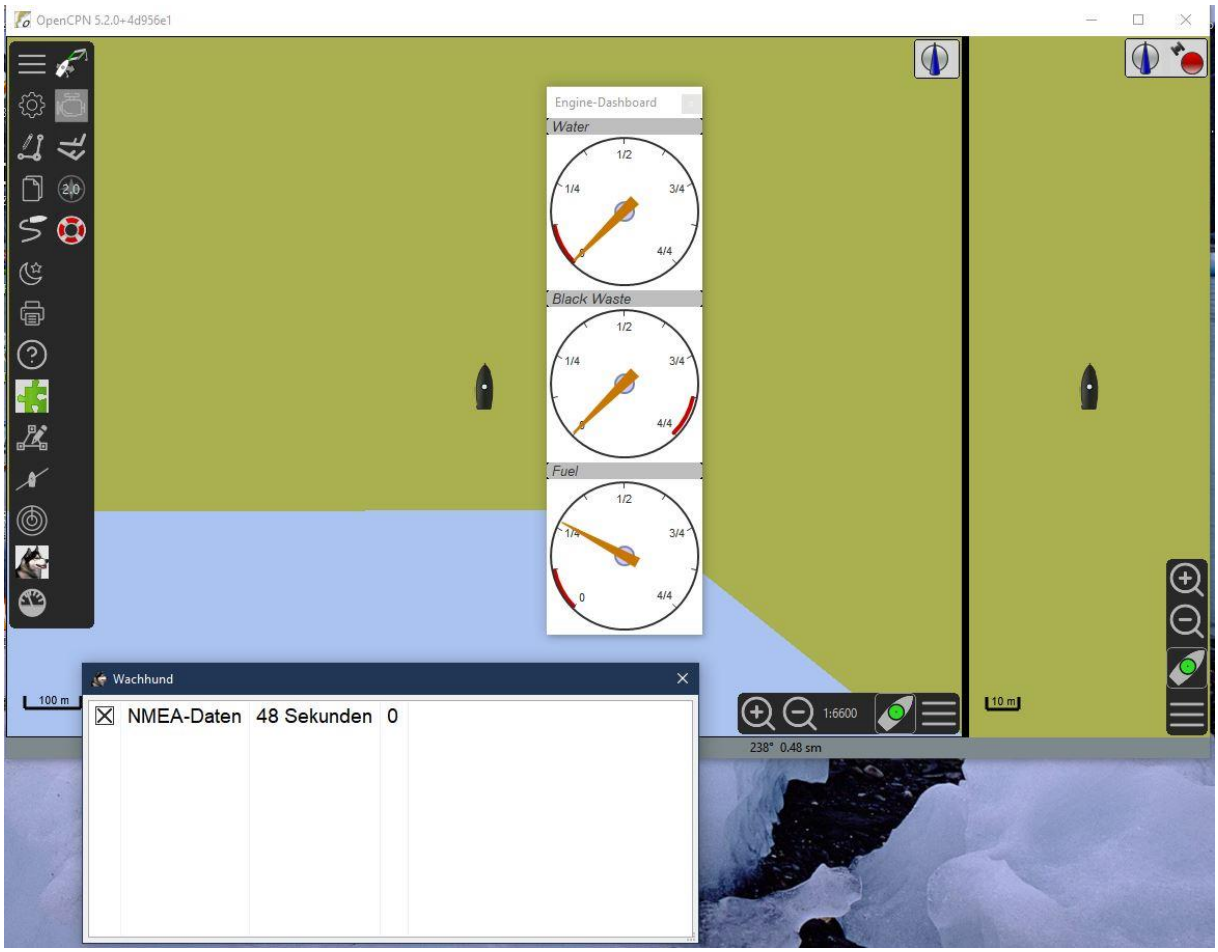

Abbildung 10 Visualisierung in OpenCPN, der Fueltank ist zu knapp 1/4 gefüllt, Watchdog zeigt, dass der letzte Datenempfang vor 48 Sekunden war.