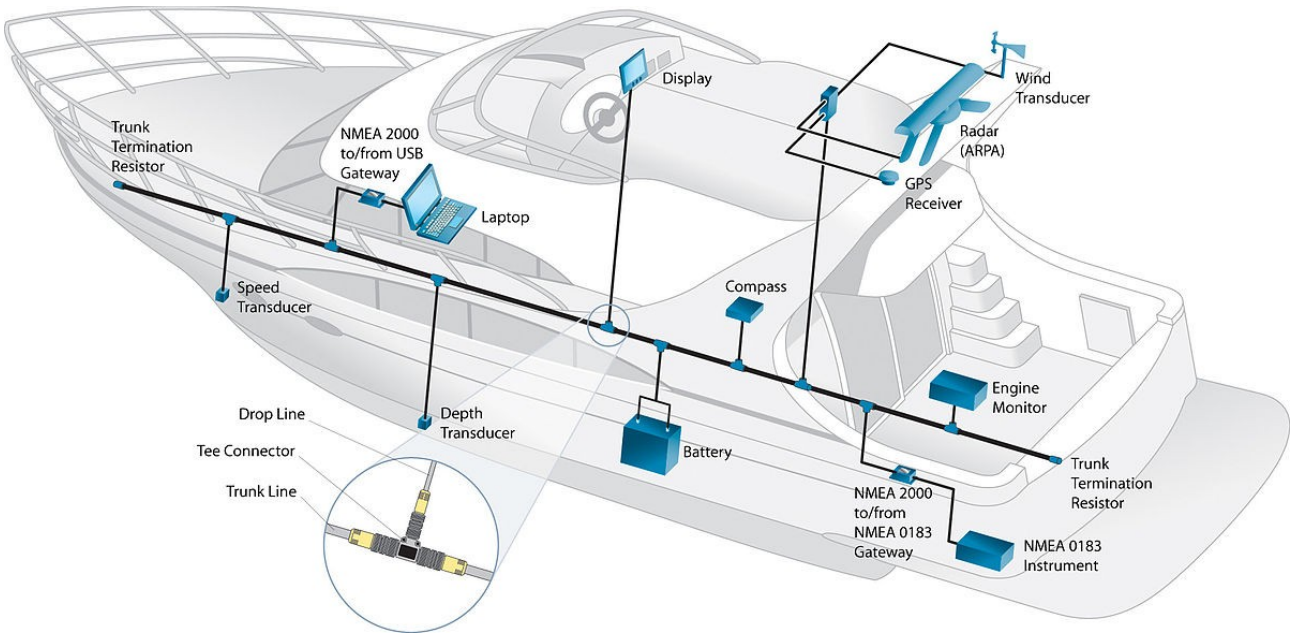


(Kilde: https://upload.wikimedia.org/wikipedia/commons/thumb/b/bf/NMEA2000_Modified_motor_yacht.jpg/1200px-NMEA2000_Modified_motor_yacht.jpg)

Guide: Integrer Arduino i et NMEA 2000 system på båten din

(Foreslå gjerne en bedre/kortere overskrift)

De fleste båter, alt fra joller til store lasteskip, bruker NMEA 2000 eller den eldre NMEA 0183 som en standardisert kommunikasjonsprotokoll mellom de ulike komponentene. NMEA-utstyr er ofte svært dyrt, så denne guiden viser hvordan du kan bruke en 30 kroners Arduino til å integrere mot dette systemet.

Innledning

Som tidligere ansatt i Kongsberg Maritime har jeg jobbet med styring og regulering av båter og skip over hele verden. Det å sitte i kapteinstolen og ha oversikt over, samt styre, enhver komponent på båten fant jeg svært interessant. Jeg var derfor klar på at da jeg i fjor kjøpte min første båt, en Quicksilver PH905, skulle den ha litt diverse leketøy slik som autopilot, ekkolodd, kart-plotter med mer. Jeg ønsket også å ha individuell styring på de to motorene, samt baugpropell, slik at jeg kunne utvikle Dynamisk Posisjonering (DP) på båten. Så vell ble det ikke, ettersom denne løsningen var svært dyr. Jeg jobber isteden med plan B – å kunne styre båten med en trådløs PlayStation kontroll. Jeg har ikke kommet helt dit enda – første steg ble å integrere en nivåsensor på vann- og septiktanken, for det var ikke et tilbehør vi kunne velge. Jeg tenkte det kanskje var flere som kunne være interessert i det samme, så jeg bestemte meg for å skrive en guide, slik at flere kan gjøre det samme.

Forord

NMEA 2000 er en lukket standard. Du kan kjøpe dokumentasjonen til en høy pris, og da aksepterer du samtidig at du ikke får lov til å fortelle andre om dokumentets innhold. All informasjon i denne artikkelen er hentet fra Timo Lappalainen som i 2015 har utviklet et eminent Arduino-bibliotek med åpen kildekode, basert på et tidligere OpenSkipper prosjekt utviklet av Dr Andrew Mason og student Jason Drake. Det er også basert på CanBoat prosjektet, utviklet av Kees Verruijt. En stor takk til disse, og spesielt takk til Lappalainen som hjalp meg å migrere koden til å fungere med Arduino Uno – ettersom han selv har skrevet koden for Arduino Mega, som ikke passer til Can-Bus shielden som jeg har brukt.

Første steg – Bestill hardware

Det første du må gjøre er å lokalisere en plass på båten med NMEA 2000 tilkopling. Hvis du er heldig, finnes det et ledig uttak. Du kan uansett begynne å eksperimentere ved å midlertidig kople fra et annet NMEA-utstyr. Hvis ikke, må du kjøpe en NMEA 2000 T-connector, f.eks. denne (https://www.maritim.no/garmin-kabel-nmea-2000-t-stykke#.WPH9yfl95hE).

Videre trenger du kabel som skal gå fra T-connectoren til CAN-BUS shielden, f.eks. denne: (https://www.maritim.no/nmea-2000-backbone-cable-2m-#.WPH_Rvl95hE). Lengden avgjør du selv, ut ifra hvor du ønsker å montere Arduinoen. Min er cirka 40cm lang. Tanken er å klippe av den ene enden, og lodde på en female DB9-connector.

Videre trenger du en microkontroller. Lappalainen har utviklet biblioteket for Arduino Due, Arduino Mega and Teensy 3.2, og han foretrekker selv Teensy 3.2 fordi den er mindre og trekker mindre strøm. Problemet med alle disse er at de har SPI-pinnene (CS, MOSI, MISO og SCK) plassert på steder som ikke er tilgjengelig for CAN-BUS shielden som jeg har brukt.

Fordelen med dem er likevel at de har vesentlig større kapasitet og minne, men for enkle oppgaver som å sende data om tanknivå fungerer Arduino Uno helt fint.

Hvis du likevel ønsker å bruke Arduino Due eller Mega kan du relativt enkelt kople SPI-pinnene på på Arduinoen (pinne 50, 51, 52 og 53) til de respektive pinnene på CAN-BUS shielden (pinne 10, 11, 12, 13). Her (https://bigdanzblog.wordpress.com/2015/01/30/cant-get-i2c-to-work-on-an-arduino-nano-pinout-diagrams/) ser du en oversikt over pinnene på de ulike mikrokontrollerene.

Du kan også finne frem loddebolten og kjøpe inn nødvendig utstyr for å kople opp etter dette skjemaet:
https://github.com/ttlappalainen/NMEA2000/blob/master/Documents/ArduinoMega_CAN_with_MCP2515_MCP2551.pdf

...eller du kan gjøre det veldig enkelt – kjøp en Arduino Uno og en Can-Bus shield, så har du alt du trenger.

Handlelisten blir dermed slik:

Arduino Uno

- F.eks: http://www.ebay.com/itm/NEW-UNO-R3-ATmega328P-CH340-Mini-USB-Board-for-Compatible-Arduino-/311155383820?hash=item48724e5e0c:g:UswAAOSwNnRYfC0R

CAN BUS Shield

- F.eks: http://www.ebay.com/itm/SPI-MCP2515-EF02037-CAN-BUS-Shield-Controller-communication-speed-high-Arduino-/401090806613?hash=item5d62e00355:g:k2cAAOSwOgdYrS2L

Female DB9-connector

- F.eks. http://www.ebay.com/itm/NEW-2set-Serial-RS232-DB9-9-PIN-Female-Assembly-Solder-Plug-Connector-with-Shell-/321842129190?hash=item4aef492926:g:kbsAAOSwu4BV2tHw

NMEA 2000 T-connector

- F.eks. https://www.maritim.no/garmin-kabel-nmea-2000-t-stykke#.WPH9yfl95hE

NMEA 2000 kabel

- F.eks. https://www.maritim.no/nmea-2000-backbone-cable-2m-#.WPH_Rvl95hE

Mini-USB kabel (Dette har du kanskje fra før)

- F.eks. http://www.ebay.com/itm/3m-Long-MINI-USB-Cable-Sync-Charge-Lead-Type-A-to-5-Pin-B-Phone-Charger-/351527269877?hash=item51d8a875f5:g:SR8AAOSwvUlWqi9M

RS232 Extension Cabel (hvis din NMEA 2000 kabel ikke er lang nok)

- F.eks. http://www.ebay.com/itm/Serial-RS232-Extension-Cable-DB9M-to-F-3m-/380314922729?hash=item588c892ee9:g:Uo4AAOSwa39Uu40N

Øvrig utstyr for å kople sensorer til Arduino.

- F.eks. Ultrasonic sensor: http://www.ebay.com/itm/1pcs-Ultrasonic-Module-HC-SR04-Distance-Measuring-Transducer-Sensor-for-Arduino-/400985326881?hash=item5d5c968521:g:kLQAAOxyNyFS-xFw
- Cat6 nettverkskabel (uten plugg) eller liknende.

Andre steg – Begynn å programmere

For å komme i gang med programmeringen, trenger du kun en Arduino av hvilket som helst slag. Hvis du har en Arduino liggende, kan du dermed begynne å skrive kode mens du venter på

leveransene i postkassa. Hvis du aldri har brukt en Arduino før, er det veldig lett å komme i gang (https://www.arduino.cc/en/Guide/HomePage).

Kople Arduinoen til PCen med en USB-kabel.

Gå deretter til følgende tre sider, og klikk på knappen **Clone or download** og deretter **Download ZIP**:
https://github.com/ttlappalainen/NMEA2000

https://github.com/ttlappalainen/NMEA2000_mcp

https://github.com/ttlappalainen/CAN_BUS_Shield

Lagre .zip-filene i en mappe du husker. Start deretter et nytt Arduino-prosjekt, og kopier innholdet av NMEA2000-master.zip/Examples/WindMonitor/**WindMonitor.ino** inn i det nye prosjektet. Lagre prosjektet et sted du ønsker å lagre koden.

Det neste som står for tur er å importere de tre bibliotekene som er linket til ovenfor. Det gjør du fra Arduino-IDEen ved å klikke på Skisse -> Inkluder Bibliotek -> **Legg til .ZIP Bibliotek...**

Videre må du, fra Verktøy -> **Kort**, og Verktøy -> **Port** velge det riktige kortet og den riktige porten for din Arduino.

Hvis alt er gjort som det skal, bør du nå kunne kompilere og laste opp koden til Arduinoen.

Tredje steg – Les av vind og temperatur, i Debug mode

Nå er tiden inne for å teste om koden fungerer som den skal. Vi trenger da et program som heter NMEA Reader. Det kan lastes ned her (http://actisense.com/products/nmea-2000/ngt-1/downloads-ngt1.html). Dette er en noe krunglete side, men du skal altså laste ned den som heter **Software Update**, med undertekst **NMEA Reader v1.xxx Setup.exe**

NMEA 2000 PGN beskjeder er rent binært og uleselig for oss dødelige. NMEA Reader er ment å integrere mot en NGT-1 som oversetter NMEA 2000 PGN beskjeder til beskjeder som vi kan lese. Det er en slik modul du kan se på forsidebildet i denne artikkelen, før laptopen.

NGT-1 kan kjøpes i Norge til den nette sum av litt over 2000 kr.
Med en Arduino til hundre kroner kan du gjøre akkurat det samme.

Illustrasjon 1: http://www.pyacht.com/images/ACT_NGT1.gif

Så, litt tilbake til valg av Arduino. Hvis ønsket ditt er å benytte kontrolleren til å overvåke alle NMEA 2000 PGN beskjeder, vil jeg nok anbefale Mega, Due eller Teensy 3.2. Hvis du bare ønsker å sende eller motta noen få sensoravlesninger, ser det ut til å holde med Arduino Uno.

For å få noe glede av NMEA Reader nå, før du har koplet systemet til båten, må du konfigurere Arduino-koden litt. Så, åpne **kopien** du har laget av **WindMonitor.ino**.

Inni setup()-blokka, fjern de to skråstrekene foran:
[code]

```
  //Serial.begin(115200);

  //NMEA2000.SetForwardStream(&Serial);

[...]

// NMEA2000.SetDebugMode(tNMEA2000::dm_Actisense);
```

[/code]

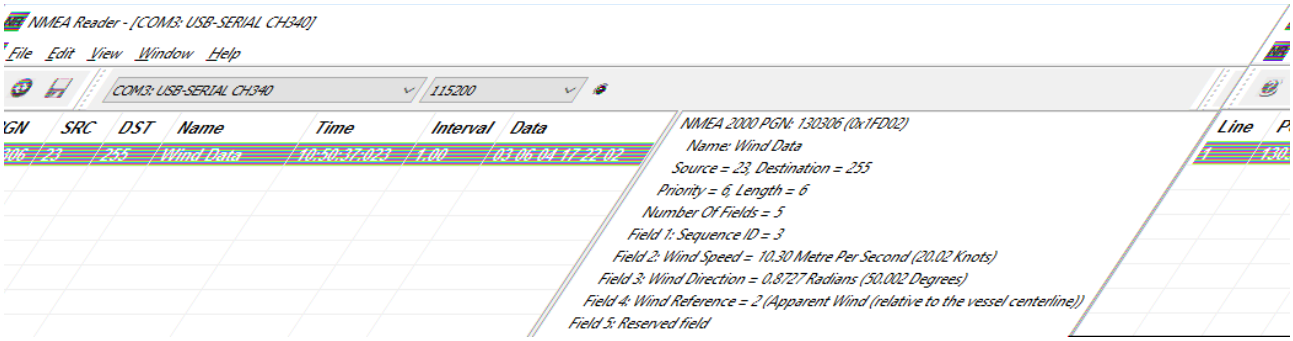
Det skulle i teorien være alt, men ettersom Arduino Uno-en har sine begrensninger i prosesseringskraft, har jeg funnet ut at følgende modifikasjoner må gjøres (merk at det kan hende at Lappalainen innen kort tid kommer til å gjøre endringer i koden slik at dette ikke lenger er nødvendig):
Legg til følgende linje under NMEA2000.EnableForward(false);

[code]

```
NMEA2000.SetN2kCANMsgBufSize(2);
```

[/code]

Du skal nå kunne åpne NMEA Reader, velge COM-port (den samme som du bruker i Arduino IDEen), og Baudrate 115200. Hvis du nå ser noe tilsvarende bildet nedenfor, er du godt i gang:

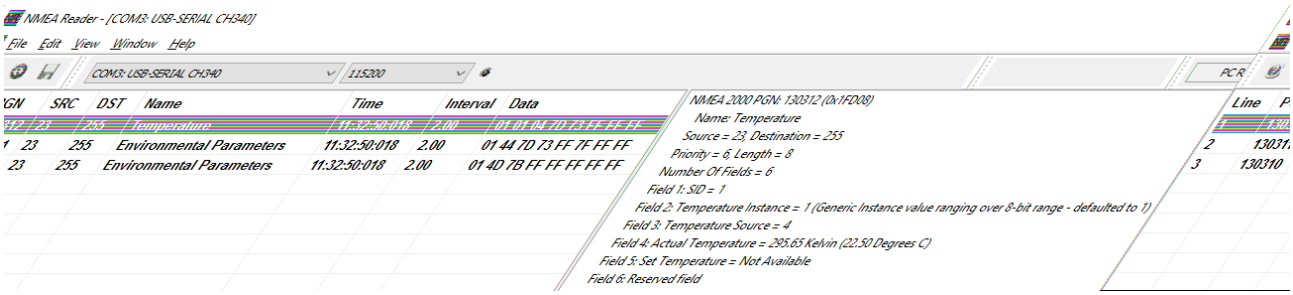


Det neste du kan gjøre er å lage et nytt prosjekt i Arduino IDEen. Denne gangen kan du kopiere over innholdet fra NMEA2000-master.zip/Examples//TemperatureMonitor/**TemperatureMonitor.ino**

NB! NMEA Reader reserverer COM-porten så lenge den er valgt. Pass derfor på å velge **– NOT SELECTED –** når du skal laste opp ny kode til Arduinoen.

TemperatureMonitor.ino prosjektet bygger på akkurat den samme logikken som **WindMonitor.ino**, så gjør akkurat de samme endringene som du gjorde med **WindMonitor.ino**.

Det du nå vil se, er at du med én Arduino kan sende data fra flere ulike sensorer samtidig. Det vil si at du nå har investert din siste krone i NMEA-utstyr, gitt at du fortsetter å utvide med sensorer som integreres mot Arduinoen. Du kan eksperimentere videre med å prøve å sende vind og temperatur sammen.



Fjerde steg – Lag det du vil

Nå som du har fått vind- og temperaturmonitorprogrammene til å fungere, er det på tide å lage noe matnyttig. Jeg har som sagt valgt å lage en nivåsensor på vanntanken min, siden det er noe jeg har savnet. Lag et nytt prosjekt som heter **FluidLevelMonitor.ino** og kopier innholdet fra din modifiserte versjon av **WindMonitor.ino**.

Bruk WinMerge (http://winmerge.org/?lang=en) eller et annet sammenlikningsprogram til å sammenlikne **WindMonitor.ino** og **TemperatureMonitor.ino**. Det vil nå være enkelt å se hvilke endringer du trenger å gjøre for å få **FluidMonitor.ino** til å fungere:

1. Beskrivelsen på første linje bør endres
2. Tallene inni TransmitMessages[] PROGMEM={} må endres. Bruk dette

(http://www.nmea.org/Assets/july%202010%20nmea2000_v1-301_app_b_pgn_field_list.pdf) dokumentet til å finne frem til at koden for Fluid Level er 127505. Du ser også de 5 parameterne den den viser.

3. Under **// Set device information** angis i alle fall Device Function og Device Class. I koden er det linket til et PDF-dokument, der du vil se at Fluid Level er device **function 150** i **klasse 75**
4. I loopen gjør du om metodekallet SendN2kWind() til **SendN2kWaterFluid()**
5. Hopp så ned til metoden **SendN2kWind()**, og endre navn på denne til **SendN2kWaterFluid().** I denne metoden ser du metodekallet **SetN2kWindSpeed.** [code]SetN2kWindSpeed(N2kMsg, 3, ReadWindSpeed(), ReadWindAngle(),N2kWind_Apprent);[/code]

   Søker du i fil-innhold etter **SetN2kWindSpeed** i hele **NMEA2000-master** mappen (som du kan pakke ut fra zip-fila), finner du at den er brukt i **N2kMessages.h**:

   [code]inline void SetN2kWindSpeed(tN2kMsg &N2kMsg, unsigned char SID, double WindSpeed, double WindAngle, tN2kWindReference WindReference) [/code]

   Da kan du videre fra denne fila søke etter **fluid** eller **127505** så vil du finne

   [code]inline void SetN2kFluidLevel(tN2kMsg &N2kMsg, unsigned char Instance, tN2kFluidType FluidType, double Level, double Capacity)[/code]

   Det vil si at metodekallet i **FluidMonitor.ino** skal endres fra **SetN2kWindSpeed** til **SetN2kFluidLevel**.

   Deretter må du få parameterne til å stemme overrens mellom **FluidMonitor.ino** og **N2kMessages.h** filene. Den første parameteren er lik. Den andre parameteren **unsigned char Instance** tilsvarer **Temperature Instance** som vi hadde i TemporatureMonitor.ino som bare var satt til 1, så vi gjør det samme her. Neste parameter er hva slags type væske som er i tanken. Som det står referert til i **N2kMessages.h** fila kan du finne en liste over de ulike typene lenger ned hvis du søker på **tN2kFluidType.** Alternativene er N2kft_Fuel, N2kft_Water, N2kft_GrayWater, N2kft_LiveWell, N2kft_Oil og N2kft_BlackWater. Den neste parameteren er **Fluid Level** som i følge **N2kMessages.h** skal være en double-verdi med tanknivået i % av full tank. **Capacity** skal være en double-verdi med tankkapasiteten i liter.

   Det blir da slik:
   [code]
   SetN2kFluidLevel(N2kMsg, 1, N2kft_Water, ReadTankPercent(), ReadTankCapacity());

   [/code]

   Metodene **ReadTankPercent** og **ReadTankCapacity** må så lages, tilsvarende de allerede

eksisterende **ReadWindAngle** og **ReadWindSpeed.** Du kan igrunn bare gi disse metodene det nye navnet. Så blir det opp til deg å lage Arduino-koden for tallet som skal returneres, istede for det som er hardkodet i eksempelet, basert sensoravlesning. For ultrasonic sensor er dette (https://github.com/ondras/arduino/blob/master/distance/Distance.ino) et godt utgangspunkt.

Femte steg – Reis til båten og kople opp

Nå gjenstår bare det morsomme – å vise dataen på Multi Funksjons Displayet (MFD) som er i båten (f.eks. en Simrad kart-plotter). Jeg nevnte tidligere at NMEA-kabelen som står på handlelisten ovenfor skal klippes av i den ene enden som ikke skal skrus fast i båtens NMEA-bus. Avisoler toppen, og det kommer da fem ledninger til syne: Svart (gnd), Rød(+12V), Hvit(CAN_H), Blå(CAN_L) og skjerm (bare en leder uten isolasjon). Skjermen kan du bare klippe vekk. Du har nå to måter å kople de 4 lederne til Arduinoen – Enten ved å skru den blå og hvite lederen til skruterminalene på CAN-BUS shielden som er merket CAN_H og CAN_L, og den røde til VIN og den svarte til GND på arduinoen. Eller den litt mer elegante løsningen – å lodde den fast til en DB9-connector plug (som angitt i handlelisten).

Se da på tegningen her (http://wiki.seeed.cc/CAN-BUS_Shield_V1.2/) på hvilke pinner du skal lodde på hvor. Husk at du skal lodde på hun-kontakten, så du må tenke speilvendt av tegningen.

Når du nå har montert CAN-BUS shielden oppi Arduino UNOen, og koplet NMEA-kabelen til NMEA-busen og CAN-BUS shielden, og koplet Arduino UNOen til din laptop, gjenstår det bare et par kodeendringer i .ino-filene du har endret.

1. **Over** linja **#include <NMEA2000_CAN.h>** legger du til:
   #define N2k_SPI_CS_PIN 10

   Dette vil gjøre at Arduinoen bruke pinne 10 som Chip Select pinne, istede for default pinne 53 som er brukt på Arduino Mega / Arduino DUE.
   MERK: Hvis du har kjøpt den originale CAN-BUS_shielden fra Seeed Studio i stede for den jeg linket til, må du kanskje bruke CS_PIN 9 istede, eller gjøre en liten loddejobb, slik det er beskrevet i http://wiki.seeed.cc/CAN-BUS_Shield_V1.2/ .

2. (Frivillig): Legg til:
   #define N2k_CAN_INT_PIN 2
   etter CS_PIN-linja. Med interrupt sikrer du at systemet ikke mister addressering og ISO request beskjeder som MFD-en bruker fra tid til annen for å sjekke at enhetene fortsatt lever.

3. Skru av debug-mode, ved å legge til de to skråstrekene du tidligere fjernet foran [code]NMEA2000.SetDebugMode(tNMEA2000::dm_Actisense);[/code]

Last opp den oppdaterte koden til din Arduino. Det skal nå, forhåpentligvis, være mulig å finne igjen verdiene blant instrumentpanelet på din MFD.

Prøv det ut, og legg gjerne igjen en kommentar nederst i artikkelen hvis du har fått det til, eller hvis du har noe spørsmål.