Chapter 1

Introduction

1.1 What is MicronetToNMEA

MicronetToNMEA is a Teensy/Arduino based project aimed at building a cheap NMEA/Micronet bridge. The initial purpose of this project was to understand Micronet wireless protocol and to be able to record wind and speed data on a PC. The understanding of the protocol went so well that MicronetToNMEA is now doing a lot more than that. It can:

- Send NMEA stream to your PC/Tablet with data on depth, water speed, wind, magnetic heading, GNSS positionning, speed, time, etc.
- Send Heading data from the navigation compass to your Micronet's displays (HDG).
- Send GNSS data to your Micronet displays (LAT, LON, TIME, DATE, COG, SOG).
- Send navigation data from OpenCPN or qtVlm to your Micronet displays (BTW, DTW, XTE, ETA).

1.2 What is NOT MicronetToNMEA

MicronetToNMEA is not waterproof and more generally not reliable. All electronics used in this project are made for hobbyist and are all but robust. In the brutal, wet and salty environment of a boat, it will likely fail quickly. So be careful that MicronetToNMEA shouldn't be used as primary navigation tool. Also note that Micronet wireless protocol has been reverse engineered and that many of its aspects are not yet properly understood. Worse, some understandings we think to be correct might very well be false in some circumstances. If you need state of the art and reliable navigation devices, just go to your nearest Raymarine/TackTick reseller.

1.3 Contributors

- Ronan Demoment : Main author
- Dietmar Warning : LSM303 drivers & Bugfixes
- Contributors of YBW forum's Micronet thread : Micronet Thread

Chapter 2

Needed hardware and software

2.1 Required hardware

To work properly, MicronetToNMEA needs at least a Teensy 3.5 board and a CC1101 based breakout board.

2.1.1 Teensy 3.5

Teensy 3.5 has been chosen as the core microcontroller of the MicronetToNMEA system. This choice has been led by one main reason : I had one available when I started investigating Micronet protocol. That's indeed a good reason but with time this board has also proven to be pretty well adapted :

- It is small
- Teensy software stack is rich and stable
- It has a lot of highly configurable peripherals
- GPIOs are 5V tolerant (important to connect 5V GNSS !)
- It has a MicroSD slot for future recording features

In theory, you can port MicronetToNMEA SW to any 32bit Arduino compatible board. Practically, this might be a different story. Several people got into troubles trying to use ESP32 boards. While this is technically feasible, Arduino's library implementation between Teensy & Esp32 board can be slightly different in some sensitive areas like interrupt handling. This makes porting complex.

Some users successfully used MicronetToNMEA on Teensy 4.0 and 4.1 boards with minimal adaptations.

Teensy boards can be ordered here : `https://www.pjrc.com/teensy/`

2.1.2 CC1101 board

CC1101 is absolutely mandatory to MicronetToNMEA. It is the IC which enable RF communication with Micronet/TackTick devices. CC1101 breakout boards are very cheap but the quality of design and components is often less than average. So do not expect to have the same distance performance than an original TackTick device. Be careful when ordering this board since it is designed for a specific range of frequencies (filter and antenna), even if the board is announced to support 434 & 868 (the IC can, but the antenna filter can not). MicronetToNMEA needs a board designed for 868/915MHz usage. Ordering the wrong board would dramatically reduce operating distance between MicronetToNMEA and TackTick devices. Here is an example of a suitable board: 868MHz CC1101

These low-cost boards are often delivered without any documentation, especially pin-out description. In that case CC1101 datasheet might help : CC1101 datasheet

2.2 Optional hardware

You can add optional HW to MicronetToNMEA to enhance its capabilities.

2.2.1 NMEA0183 GNSS

If you want to connect a GNSS/GPS to MicronetToNMEA, there is only one important point : it must output localization data on a RS232 link using NMEA0183 format. An example of cheap GNSS which fits the need is the UBLOX NEO-M8N. The NEO-M8N can directly output NMEA stream to its serial output. Avoid too cheap offers from unknown HW sources, this might be counterfeit hardware.

2.2.2 LSM303DLH or LSM303DLHC navigation compass breakout board

Connected to Teensy I2C bus, this IC will allow getting magnetic heading. MicronetToNMEA automatically detect the presence and type of LSM303DLH/DLHC on its I2C bus.

2.2.3 HC-06 Bluetooth transceiver

You can connect HC-06 device to MicronetToNMEA serial NMEA output to easily get a wireless connection to a PC/Tablet. Note that MicronetToNMEA does not configure HC-06 link, it is up to you to configure HC-06 before connecting it.

2.3 Required software

2.3.1 Arduino IDE (required)

Arduino IDE provides gcc-arm compiler and all libraries necessary for MicronetToNMEA. This is the first software you must install.

2.3.2 Teensyduino (required)

Teensyduino is an extension to Arduino IDE which add full support to all Teensy's board, including Teensy 3.5. It must be installed on top of Arduino IDE to enable compilation for Teensy 3.5.

2.4 Optional software

2.4.1 Sloeber (optional)

If you plan to do more than just compile MicronetToNMEA's code, you probably need a more serious IDE. Sloeber is an Arduino compatible version of Eclipse. It provides many useful features which will highly improve your productivity. It requires Arduino IDE and Teensyduino to be already installed.

Chapter 3

Compilation

3.1 With Arduino IDE

Here are the steps to compile MicronetToNMEA with Arduino IDE:

- Get the source code from MicronetToNMEA repository (`https://github.com/Rodemfr/MicronetToNMEA`)
- Double-click on MicronetToNMEA.ino. This should open Arduino IDE.
- In Arduino IDE, select Teensy 3.5 target HW with menu “Tools->Board->Teensyduino->Teensy3.5”
- Go to menu “Tools->Manage Libraries...” and install the following libraries : SmartRC-CC1101-Driver-Lib and TeensyTimerTool
- Click on “Verify” button in the button bar, this should compile the project without error.
- Connect your Teensy 3.5 board onto USB port of your PC and Click “Upload” button to upload MicronetToNMEA binary into Teensy flash memory

3.2 With Sloeber

Here are the steps to compile MicronetToNMEA with Sloeber IDE:

- Before trying to compile with sloeber, you must have successfully compiled with Arduino IDE
- Start Sloeber and create your Workspace as requested Select menu “File->New->Arduino Sketch"
- In Sloeber, select menu Arduino->Preferences
- Add Arduino’s library and hardware path in the path lists
- Exit the panel by clicking "Apply and Close"
- Select menu File->New-Arduino Sketch
- Name your project "MicronetToNMEA"
- Don’t use default project location and set the location to your git cloned repository of MicronetToNMEA
- Click "Next"
- Select Teensy’s platform folder in the corresponding drop down menu
- Select "Teensy 3.5" board
- Select "Faster" optimization
- Select "Serial" USB Type
- Select 120MHz CPU Speed
- Click "Next"

- Select "No file" as code

Your project should be compiling now.

Note that Sloeber can be somewhat picky with toolchain or library paths. So don't be surprised if you have to handle additional issues to compile with it. The effort is worth, code productivity with Eclipse is way beyond Arduino IDE.

3.3 Compile time configuration

By default, MicronetToNMEA is configured for a specific HW layout. This means that it is configured to be connected through specific SPI, I2C or GPIO pins to various boards. This configuration can be changed to some extent to adapt your own needs. The file bearing this configuration is "BoardConfig.h".Note that no coherency check are made in the software. It is your responsibility to provide a reachable configuration (i.e. not to connect SPI wires to non SPI capable pins). Table 3.1 lists all available switches and their meaning.

FREQUENCY_SYSTEM	Defines which frequency range is used by your Micronet network (0=868MHz, 1=915MHz)
NAVCOMPASS_I2C	Sets the I2C bus to which the navigation compass (i.e. LSM303DLH(C)) is connected. Defined as per "Wiring" library definition (Wire0, Wire1, etc.)
RF_SPI_BUS	Defines SPI controller connected to RF IC (SPI, SPI1, SPI2)
RF_CS0_PIN	Defines SPI Chip Select line connected to RF IC
RF_MOSI_PIN	Defines MOSI pin of SPI bus connected to RF IC
RF_MISO_PIN	Defines MISO pin of SPI bus connected to RF IC
RF_SCK_PIN	Defines SCK pin of SPI bus connected to RF IC
RF_GDO0_PIN	Defines GDO0 pin of SPI bus connected to RF IC
LED_PIN	Defines the pin driving the LED, which is used for error signaling
GNSS_UBLOXM8N	Enable automatic configuration of UBLOX M8N GPS (0=disabled, 1=enabled)
GNSS_SERIAL	Defines on which serial port is connected the NMEA GNSS (Serial, Serial1, Serial2, etc.)
GNSS_BAUDRATE	Defines GNSS bitrate in baud
GNSS_CALLBACK	Defines the name of the callback function called when new bytes arrive on the configured serial port
GNSS_RX_PIN	Defines serial RX pin connected NMEA GNSS TX pin
GNSS_TX_PIN	Defines serial TX pin connected NMEA GNSS RX pin
USB_SERIAL	Defines which serial port is connected to USB serial converter
USB_BAUDRATE	Defines baud rate of USB serial converter
CONSOLE	Defines on which serial port is displayed the console (can be USB_CONSOLE or WIRED_SERIAL). Can be on the same serial link than NMEA_IN and NMEA_OUT
NMEA_OUT	Defines on which serial port to output NMEA stream. (can be USB_CONSOLE or WIRED_SERIAL). Can be on the same serial link than CONSOLE and NMEA_IN
NMEA_IN	Defines on which serial port to read input NMEA stream. (can be USB_CONSOLE or WIRED_SERIAL). Can be on the same serial link than CONSOLE and NMEA_OUT

Table 3.1: Configuration switches in BoardConfig.h

Chapter 4

Installation

Teensy board must be connected to other boards with the same scheme than you have defined in “BoardConfig.h”. No check is made by MicronetToNMEA software to verify that your configuration is matching your actual connections. You must carefully verify that you properly connected the various devices since a wrong connections can possibly damage your hardware, especially with respect to power supply connections which are mixing 3.3 & 5V levels.

4.1 Power supply

The first and most important connection to build is the power supply. You have two options there, you can either :

- Power the system via USB
- Power the system using external DC power source

4.1.1 Power via USB

This is the most straightforward way to power the system : just plug an USB cable in the Teensy connector and it will be powered by the connected PC. Teensy board is equipped with a voltage regulator which provides 3.3V. This 3.3V voltage can be used to power other boards of the system. Be careful that USB 2.0 limits 5V output current to 500mA, but you should be even more careful since Teensy’s regulator recommends not to exceed 250mA for 3.3V. So you must take care that your system does not exceed these limits. As an example, table 4.1 shows maximum current values for various boards.

Board	Voltage source	Max current	Comment
Teensy 3.5	3.3V	50mA	CPU ruuning at 120MHz
CC1101	3.3V	40mA	RF at 868MHz
NEO M8N GNSS	5V	45mA	M8N is 3.3V but the board is 5V
LSM303DLH(C)	3.3V	10mA	Unspecified in datasheet, value assumed
HC06 Bluetooth transceiver	3.3V	40mA	Peak during pairing

Table 4.1: Current consumption of typical boards

USB powering is especially useful when you plan to output NMEA through USB-serial. In this case, the connected PC/Tablet will provide power to the system and when MicronetToNMEA is not needed anymore, just unplug the USB cable.

4.1.2 Power with an external DC source

While USB powering is easy to setup, it not a common source of power in a boat. It is more usual to get two wires with an unstable battery voltage between 11V and 15V. In that case, you will need a voltage regulator or a DC-DC converter which will be used to produce a stable 5V for the system. This 5V source can then be connected to the Vin pin of Teensy 3.5. The on-board regulator will then produce 3.3V from this input. When Vin pin is connected to an external source of power, you must not connect an USB cable to avoid short circuit between Vin and Vusb which are connected together on Teensy by default. The Vin pin can

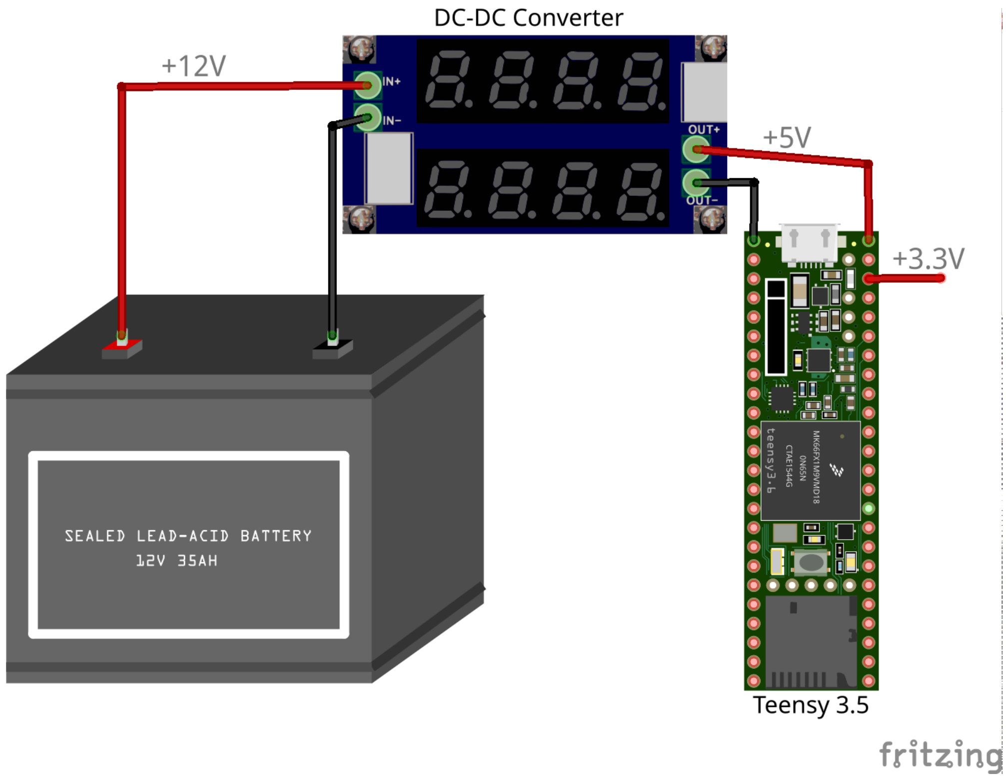

Figure 4.1: Powering Teensy with a DC-DC converter

handle voltages from 3.6 to 6V but it is strongly recommended to use 5V here. This way, if you accidentally connect a USB cable while powering Vin, there will be no short circuit.

4.2 Connecting CC1101

CC1101 uses 3.3V voltage so you can connect Teensy's 3.3V & GND pins to CC1101's VCC & GND. MOSI(SI), MISO(SO), CS0 and GD0 must be connected as per your BoardConfig.h definitions. Note that GD2 isn't used by MicronetToNMEA and doen't need to be connected to Teensy. Figure 4.2 shows how to connect CC1101 with the default configuration.

4.3 Connecting LSM303

LSM303DLH(C) uses 3.3V voltage so you can connect Teensy's 3.3V & GND pins to CC1101's VCC & GND. In addition SDA & SCL must be connected as per your BoardConfig.h definitions. Note that DRDY, I1 & I2 don't need to be connected. Figure 4.3 shows how to connect LSM303DLH(C) with the default configuration.

4.4 Connecting GNSS

Unlike CC1101 or LSM303DLH(C), GNSS/GPS boards are often requiring 5V VCC as power. So you have to connect it directly to DC-DC Converter's output. You should check however that you GNSS board is not 3.3V powered, in which case you should use one of Teensy 3.3V pin. TX and RX pins must then be connected respectively on RX and TX of Teensy's UART. Noyte that Teensy 3.5 is 5V tolerant, so you connect GNSS even if it is using 5V output. Figure 4.4 shows how to connect GNSS for the default configuration.

MicronetToNMEA can connect to a wide variety of GNSS. You just have to configure the GNSS with the correct parameters before connecting it. GNSS has to output an NMEA compatible stream at the same bitrate than specified in BoardConfig.h. MicronetToNMEA can automatically configure the GNSS if it is a UBLOX Neo M8N. Just enable GNSS_UBLOXM8N option in BoardConfig.h.

GNSS should output the following sentences :

- GGA : Position

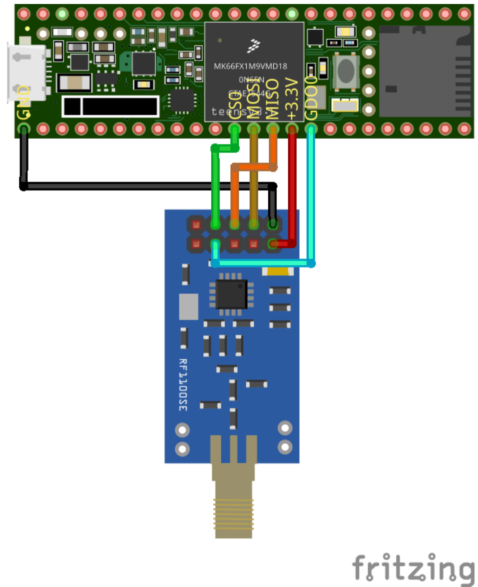

Figure 4.2: Connecting Teensy and CC1101

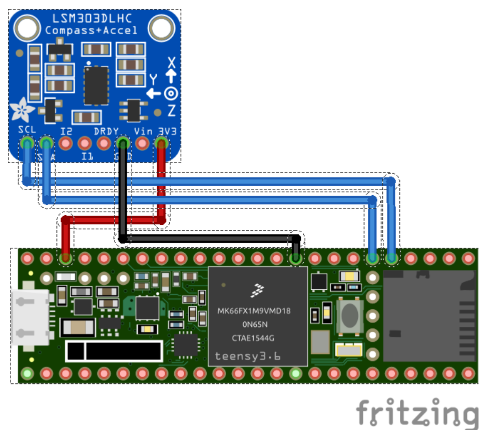

Figure 4.3: Connecting Teensy to LSM303

- RMC : Time
- VTG : Track and speed

4.5 Connecting HC-06 modules

When MicronetToNMEA is configured to send its console and/or NMEA output to a standard wired UART (i.e not USB), you can consider connecting a HC-06 Bluetooth transceiver to easily get wireless connectivity to your PC/Tablet. HC-06 is powered with 5V but can handle 3.3V signals. Only VCC, GND, RXD & TXD need to be connected. Figure 4.5 shows how to connect HC-06 with the default configuration.

As for the GNSS, MicronetToNMEA does not configure HC-06 itself. It is your responsibility to configure HC-06 properly (i.e. with parameters matching BoardConfig.h) prior to connecting it to Teensy.

4.6 Recommandations

4.6.1 RF performance

Micronet devices cannot legally exceed a handful of milliwatts of transmit power. As a consequence the reception range will hardly exceed 10-15m. It is important not to put metal objects or panels between your various devices. This would dramatically reduce reception range of MicronetToNMEA. You should also be

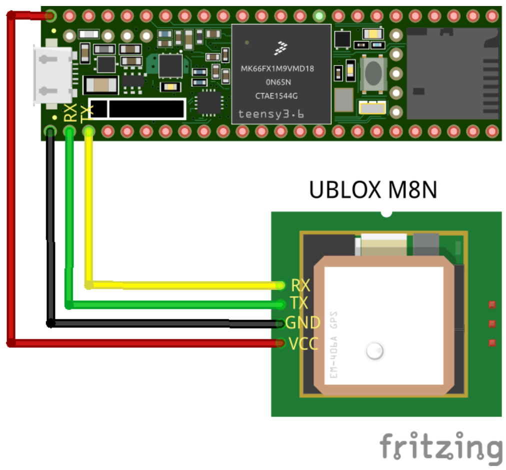

Figure 4.4: Connecting Teensy and GNSS

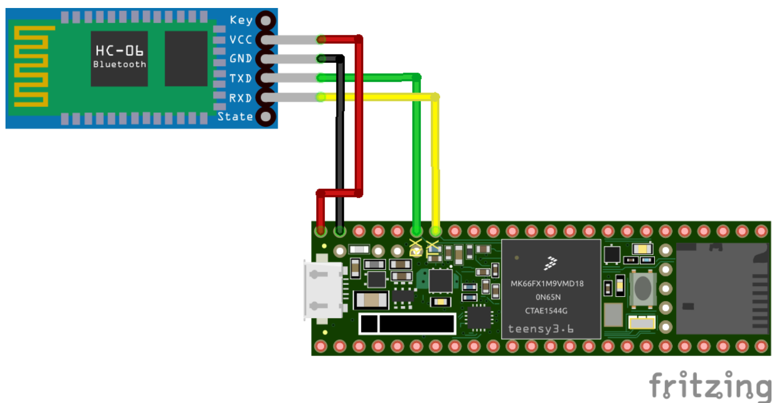

Figure 4.5: Connecting Teensy and HC-06

careful if you are racing and using carbon sails. If the sails are between your wind vane and MicronetToNMEA, you will likely not receive wind data. Carbon disturbs RF transmissions. It is generally considered not to be a good idea to use wireless electronics when you have a carbon boat or carbon sails. Fiberglass doesn't attenuates the signal (only a little), so it is safe to attach your device inside a cabin in a GRP boat.

4.6.2 Magnetic compass disturbances

If your are using LSM303 to get magnetic heading, it is very important to put your MicronetToNMEA assembly far from other electrical devices, especially those carrying a bit of power. This can highly disturbs magnetic compass. As an example, a running 24" TV monitor can deviate the compass by 20° at 50cm and still a few degrees at 1m. Also, metal must be avoided. It deviates magnetic field. The bigger the piece of metal is the farthest you should put your device. In a boat you should avoid the keel, batteries and the inboard engine.

4.6.3 Magnetic compass calibration

An electronic compass must be properly calibrated to produce good values. The theory tells us that a perfect calibration would need to have your device attached in its final position and to spin your boat around the 3 axis. This is generally not something we want to do. To achieve a good calibration with less complex operations, the best is to calibrate MicronetToNMEA outside your boat, far from any metal or electronic device. You should also keep it as far as possible of its power supply. Once calibrated, get it back to your boat and fix it in a proper place. This should give correct results.

Chapter 5

Usage

- Scanning Micronet networks
- Attaching MicronetToNMEA to your existing Micronet networks
- Calibrating RF frequency
- Calibrating navigation compass
- Starting NMEA conversion

Chapter 6

NMEA

- Supported sentences (IN and OUT)