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

Required and optional hardware

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

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, making porting complex. Teensy boards can be ordered here : `https://www.pjrc.com/teensy/`

2.1.2 CC1101 board

These boards are very cheap but the quality of the design and components is often 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 board cannot). MicronetToNMEA needs a board designed for 868MHz usage. Ordering the wrong board would dramatically reduce operating distance between MicronetToNMEA and TackTick devices. Example of suitable board: 868MHz CC1101

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 its data to the 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. Be careful however to ensure that the model youorder is not counterfeit and really has flash memory to save its configuration. Avoid too cheap offers from unknown HW sources.

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(c) 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.

Chapter 3

Required and optional software

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.

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.

3.3 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 4

Compilation

4.1 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

4.2 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 ort 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.

4.3 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 GPIOs 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 checks are done in the software. It is your responsibility to provide a reachable configuration (e.g. not to connect SPI wires to non SPI capable pins).

Here is the description of its configuration switches:

- 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_SERIAL: Defines on which serial port is connected the NMEA GNSS (Serial, Serial1, Serial2, etc.).
- GNSS_BAUDRATE: Defines on which serial port is connected the NMEA GNSS.
- 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

Chapter 5

Installation

Teensy board must be connected to other boards with the same scheme than you have defined in "Board-Config.h".

- How to connect
- How to supply power
- Using console
- Configuring GNSS
- Configuring HC-06 Bluetooth HW

Chapter 6

Usage

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

Chapter 7

NMEA

- Supported sentences (IN and OUT)

Chapter 8

Future

- Power consumption