LTM driven Artificial Horizon

MIT License

Copyright (c) 2023 altermac

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Inhaltsverzeichnis

Introduction

When you fly FPV with an Cockpit-Camera you might wan't some eyecandy on the dashboard of your plane. With this project I use a Generic NodeMCU ESP8266 to display an artificial horizon on a 0.96“ SSD1306 OLED display. Both components together need only a 5V power supply with low current.

LTMReader Class is based on parts of ltm_telemetry_reader of Paweł Spychalski. (originaly found at https://github.com/DzikuVx/ltm_telemetry_reader).

Hardware you need

One of the following development boards will fit your needs:

- NodeMCU ESP8266 development board

The ESP8266 boards are powerful small computer with Wifi-Option. The project does not need Wifi, but these boards can also be used for other purposes. The i2c bus must be supported by the board manager of Arduino IDE. Some ESP8266 with onboard OLED have non standard I2C-Pins and do not work with the Adafruit SSD1306 library used by this project. Prefer boards without soldered Dupont pins if you want to solder cables yourself.

You will need an OLED Display:

- 0.96 inch OLED display for i2c

There are several colors available. I prefer blue or white single colour displays, an OLED with two colours will have 8 or 10 lines in yellow and the rest of the display is blue or another colour. The colour can't be changed, the display only supports 1bit color. On or off.

Price for complete hardware for this project should be around 10$ to 25$ from local shops, amazon or via China direct orders. Not including your PC/Mac, an USB cable, a working flight controller and some wire.

Software you need

The project uses Arduino IDE. This is available here: https://www.arduino.cc/en/software

If you are using Arduino IDE for the first time then you have to set up your preferences. Go to „file“>“Preferences“:

First set your sketchbook location. You have to put the downloaded sketch folder of my project into the folder you set here.

Second add additional Boards Manager URLs at the bottom of the Preferences Window. For ESP8266 you can use one of the following URLs:

- https://arduino.esp8266.com/stable/package_esp8266com_index.json
- https://raw.githubusercontent.com/SpacehuhnTech/arduino/main/package_spacehuhn_index.json

Both URLs have board packages for the hardware mentioned above.

Before you connect your board go to the Boards Manager and install one of the listed Board Managers that lists your development boards (ESP8266).

Go to the Library Manager and install the following Libraries:

- Adafruit GFX Library
- Adafruit SSD1306

Get the Arduino sketch

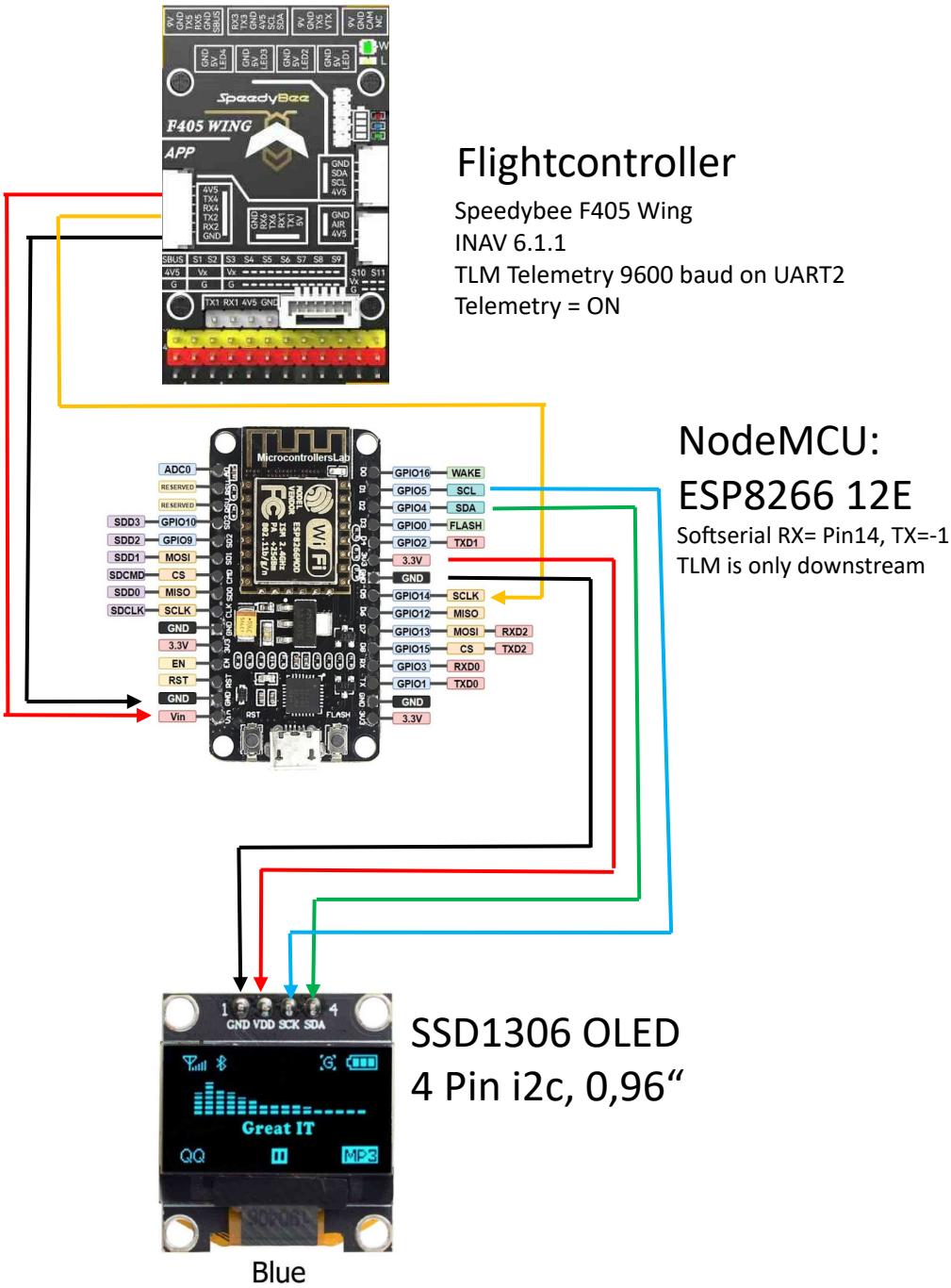
Go to https://github.com/altermac/LTM-driven-Artificial-Horizon

If you read this you might have been there before. download the code as zip-file. There is a folder **Artificial-Horizon** inside. Put this folder in your local Sketch folder of the Arduino IDE (see above). Open the Sketch in Arduino IDE.

Connect your development board via USB to your PC or Mac and choose the right port an board. Next you upload the Sketch and you are nearly done.

Some wiring to do

But the opera does not end without the fat lady singing: NodeMCU and a SSD1306 OLED will add around 15g without extended wiring to your model aircraft. But it has some features to make it a valuable external processor for our flightcontroller.



NodeMCU ESP8266 development boards have their own bec. They can be powered from an external source and can power it’s own external devices. Most boards accept Voltages from 3.3 to 9 Volts, so you can use a 9V Block battery or even a 1S Lipo to power your application. In the diagram above I use the 5V output provided by the flight controller.

And there is a Wifi module on board, wich could be used for an (inofficial) OpenID transmitter. So you get the chance to use it as a lost model finder. But with this project the Wifi device is not used.

With a NodeMCU development boards you will need the following connections :

- 3 wire from flightcontroller to your development board:
  - + 5V
  - GND
  - TX line of the chosen uart
- 4 wire connection from development board to SSD1306 OLED
  - + 3.3V
  - GND
  - SDA
  - SCL

Look at the picture above for the correct pins and check if they match to your board.

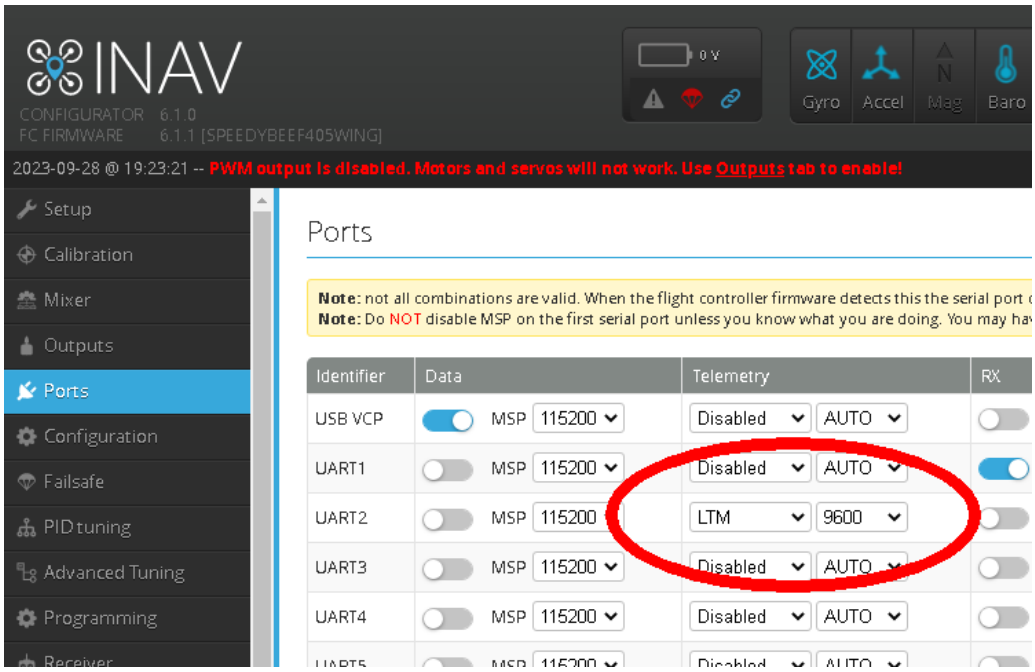
Since flightcontroller and development board can connect to USB, there could be trouble when you connect development board and flight controller via USB. So you should connect flight controller OR development board to USB. Never connect both. The development board uses an internal BEC to deliver 3.3V that can be damaged with two power sources.

The flight controller will update the TLM port less frequent on startup. After some time the TLM updates will come faster.

I have tested this setup with INAV 6.1.1 but Ardupilot and even Betaflight will support the TLM protocol as part of their receiver telemetry. I used a Speedybee F405 Wing i had lying around. Any other flight controller will do.

Settings for inav 6

At first go to the ports tab in inav configurator and look for a free UART. The UART TX pin on the flight controller should be available to connect to the NodeMCU. Set the UART speed to 9600.



I use a Speedybee F405 Wing for desktop testing. This flight controller has a dedicated Telemetry connector for UART2 and UART4 wich also provides +5V and Ground. Next you have to activate Telemetry on the Configuration Tab:

Thats all for INAV.

No C++ programming needed?

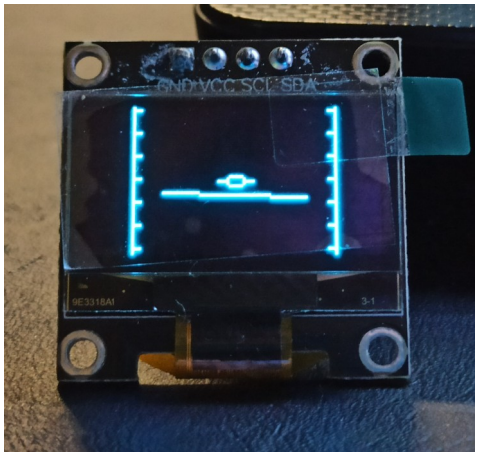
If your development board uses another pin or you want to change the type of dashboard or artificial horizon you only have to edit one file: artificial-horizon.ino.

This file has a definition for the pin in line 4 and some calls . Uncomment this line and set the number of the GPIO-Pin you want to use.

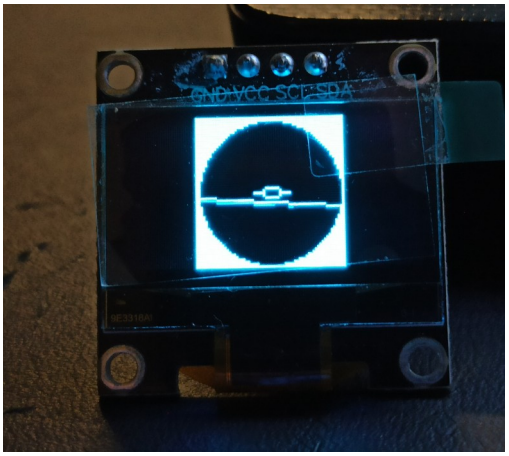
You can choose one of four different designs:

```
// choose one of the following 4 lines by removing comment:
ArtHorizon.hud_horizon(TData.roll, TData.pitch); // western:HUD-Design
//ArtHorizon.art_horizon(TData.roll, TData.pitch);  // western: spheric design
//ArtHorizon.soviet_horizon(TData.roll, TData.pitch);  // soviet: moving plane
//ArtHorizon.dashboard(TData);  // Dashboard with western artificial horizon
```

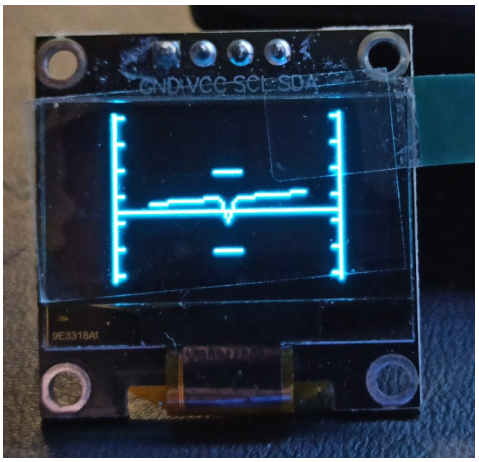
Remove the Slashes (“//”) in front of one line and comment any other active line of those four lines with //. The OLED will look like this: (all pictures done with nearly same pitch and roll)



hud_horizon



art_horizon



soviet_horizon

dashboard

The call for ArtHorizon.init() has two optional parameter:

init(int pinSDA = SDA, int pinSCL = SCL).

You can define alternative pins for SDA and SCL. This is only necessary when your development board has a faulty definition for SDA and SCL with the Arduino Board Manager. I needed this for an “Ideaspark ESP8266 0.96’’ OLED development board”, which has a quiet odd design.