LTM driven Artificial Horizon (ESP32C3)

MIT License

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 ESP32C3 Supermini development board 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 the principles 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:

- ESP32C3 development board
- ESP32S3 development board

The ESP32 boards are powerful small computer with Wifi-Option. They come in different packages and with different boards and pinouts. **Please refer to the documentation of your specific board for i2c-Pins (SDA/SCL) and available Uart1/Serial1 pins (TX/RX).** I use “ESP32C3 Dev Module” from the Espessif esp32 board manager package. There are development boards from 16 pins up to 36 pins available and they all use different pin definitions for uart and i2c. The project does not need Wifi, but these boards can also be used for other purposes. 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.

Before you connect your board go to the Boards Manager, search for ESP32 and install the “esp32” package from Espressif Systems.

Go to the Library Manager and install the following Libraries:

- Adafruit GFX Library
- Adafruit SSD1306

Get the Arduino sketch

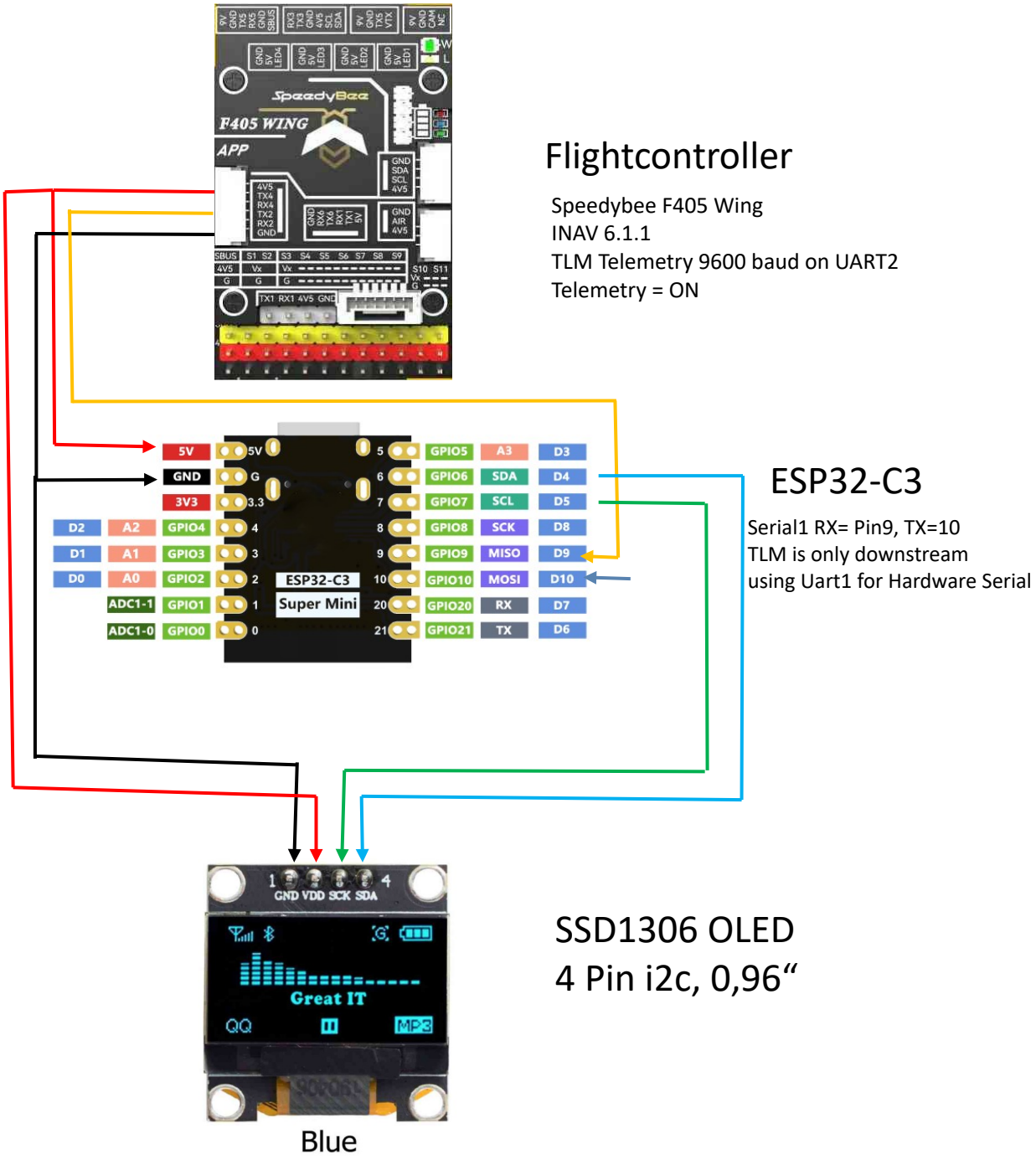
Go to https://github.com/altermac/LTM-driven-Artificial-Horizon

If you read this you might have been there before. download the code as zip-file. There is a folder **Artificial-Horizon-ESP32C3** 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 and development board. Next you upload the Sketch and you are nearly done.

Some wiring to do

ESP32C3 and a SSD1306 OLED will add around 7g without extended wiring to your model aircraft. But it has some features to make it a valuable external processor for our flightcontroller.



In this diagram i Use an ESP32C3 Supermini development board. It has only 16 pins and **no onboard BEC**. In the diagram above I use the 5V output provided by the flight controller to power the ESP32C3 Supermini and the 0.96” OLED.

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 :

- 4 wire from flightcontroller to your development board:
  - + 5V
  - GND
  - TX line of the chosen uart
  - RX line of the chosen uart
- 4 wire connection to SSD1306 OLED
  - + 5V (coming from flightcontroller)
  - GND (coming from flightcontroller)
  - SDA (coming from ESP32C3)
  - SCL (coming from ESP32C3)

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

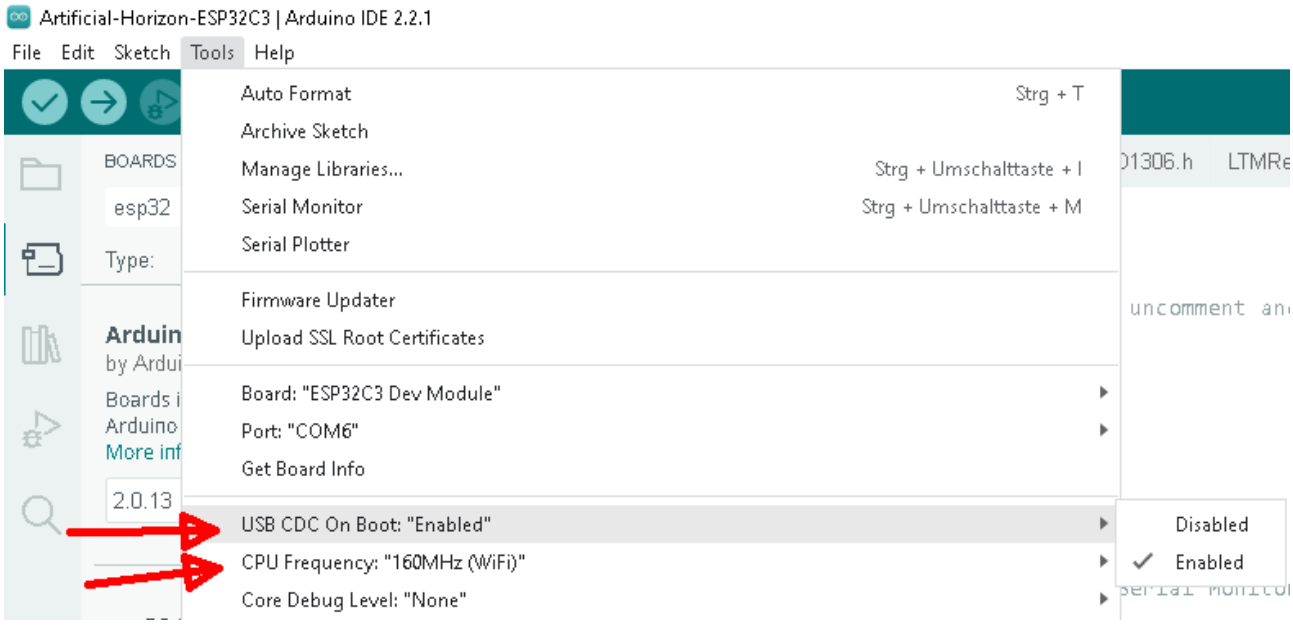
Since flightcontroller and development board can connect to USB, there could be trouble when you connect development board and flight controller via USB. So you should connect flight controller OR development board to USB. Never connect both. The development board can be damaged when using two power sources in parallel.

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.

Board Settings in Arduino IDE

As we are using the Hardwareserial, and need some performance, there are two settings to do:



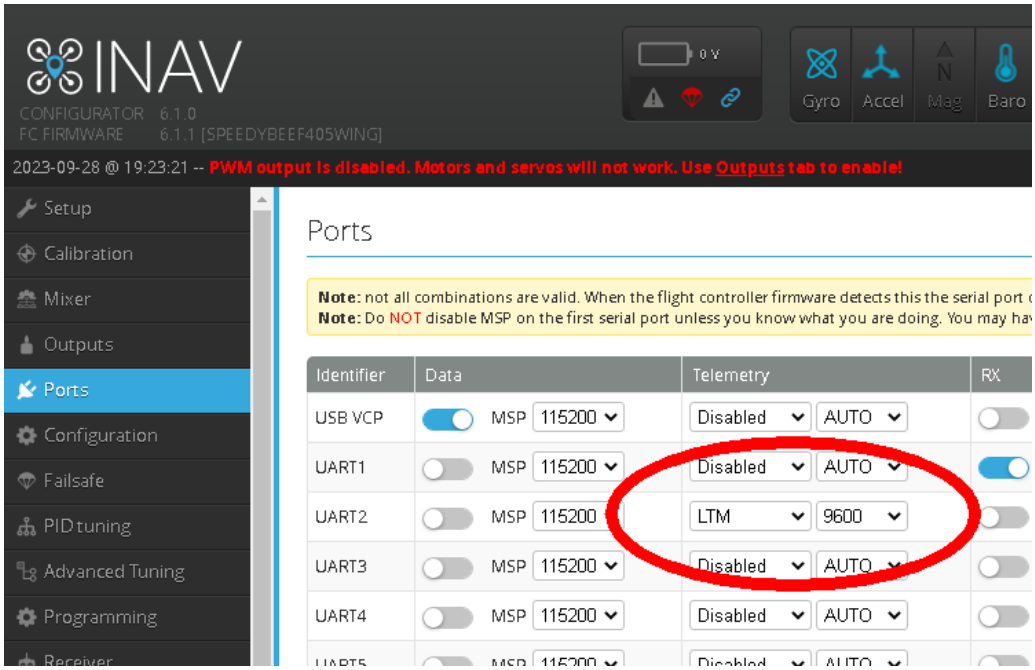
First Set “USB CDC On Boot” to enabled. This will start the Hardware CDC Serial controller on ESP32C3 and ESP32S3. This one is way faster than any Softserial and the will lead the Serial.printf-commands to uart0, wich is provided to your PC via USB or RX/TX-Pins of your board. We cannot use uart0 for anything else in this project.

Uart1 will be free as Serial1 to use for the flightcontroller-tlm-connection.

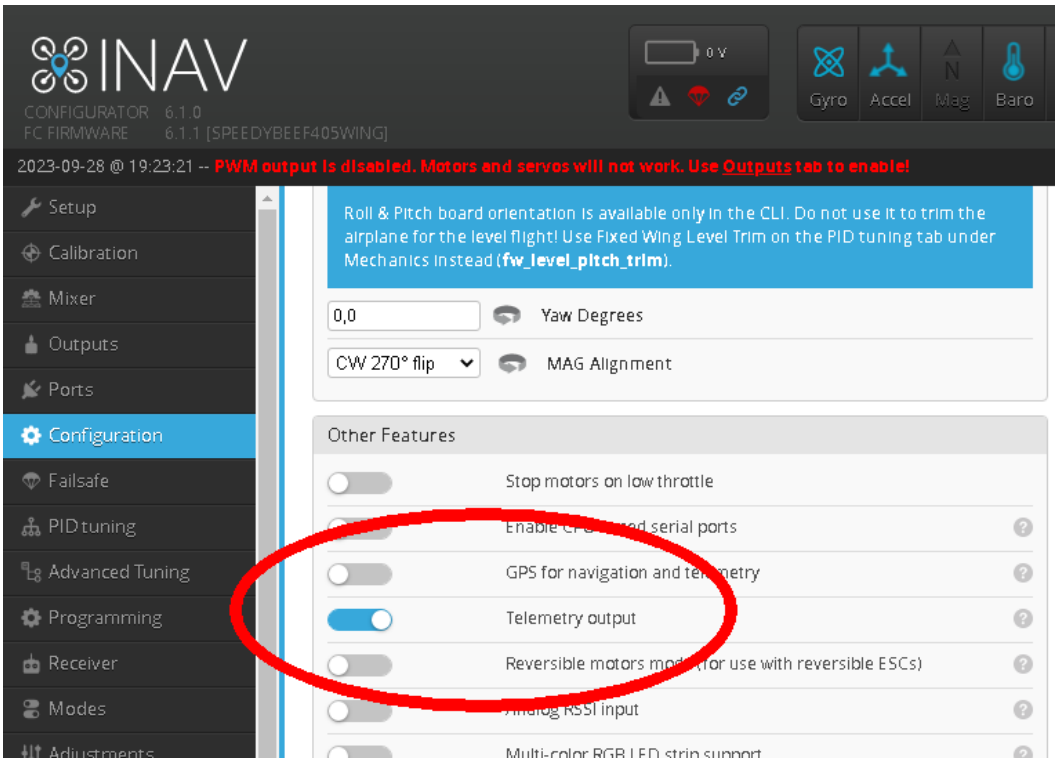
Second setting is the CPU Frequency: 80 or 160MHz will speed up things. The artificial horizon will have no visible lag or latency after that.

Settings for inav 6

At first go to the ports tab in inav configurator and look for a free UART. The UART RX/TX pins on the flight controller should be available to connect to the ESP32C3. 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.

As there are many different board designs based on ESP32C3 / ESP32S3, the default package for the boards manager often contains the false pins for SDA/SCL and Serial1. The correct pins can be seen in the documentation of your development board or on the pinout diagram that comes with your order. Most chinese vendors provide the correct pinout with their item description when offered on Amazon or Alibaba. All settings are made in Artificial-Horizon-ESP32C3.ino.

Set your pins for the serial connection to the flightcontroller here:

```
 // Telemetry initialize
 TData.init(10,9); // without parameter default is RX=10 (GPIO10),TX=9 (GPIO9)
```

Set the correct SDA and SCL here:

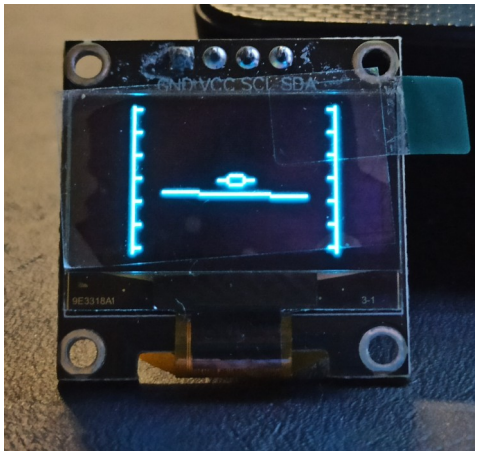
```
// Articicial Horizon
  ArtHorizon.init(6,7); // use without parameter when SDA and SCL are set correct by boards manager
```

You can choose one of four different designs by comment and uncomment:

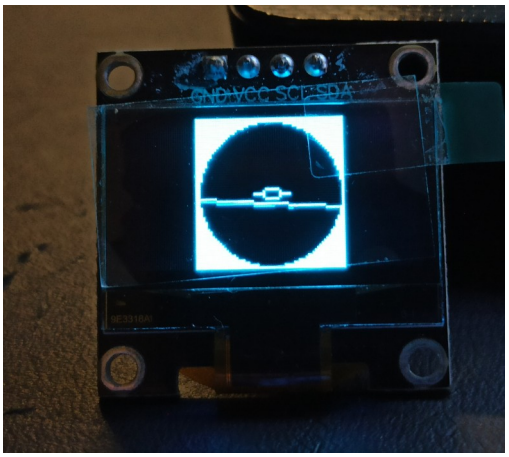
```
// choose one of the following 4 lines by removing comment:
ArtHorizon.hud_horizon(TData.roll, TData.pitch); // western:HUD-Design
//ArtHorizon.art_horizon(TData.roll, TData.pitch);  // western: spheric design
//ArtHorizon.soviet_horizon(TData.roll, TData.pitch);  // soviet: moving plane
//ArtHorizon.dashboard(TData);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

Remove the Slashes (“//”) in front of one line and comment any other active line of those four lines with //.

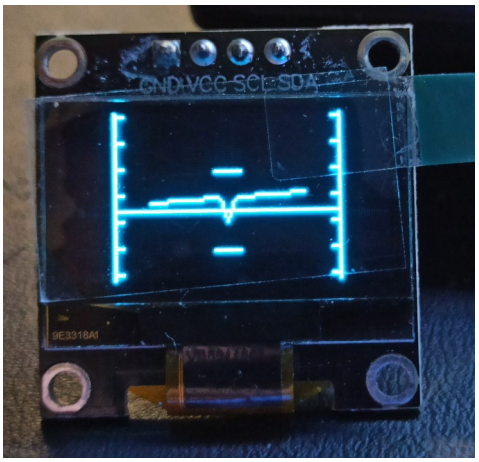
The OLED will look like this: (all pictures done with nearly same pitch and roll)



hud_horizon



art_horizon



soviet_horizon

dashboard