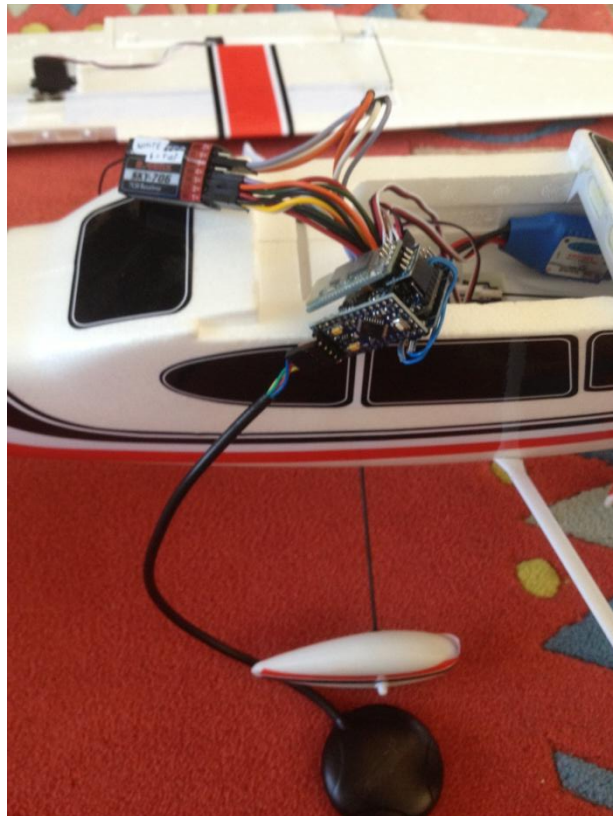


How to connect GPS via I2C with Multiwii and mMWC Shield

By Carolyn Swinney

Created 5th August 2013



Disclaimer – all the content in this How to Guide is for informational purposes only. The author makes no representations as to the accuracy or completeness of any information provided below or found by following any link in this file. The author will not be liable for any errors or omissions in this information nor for the availability of this information. The author will not be liable for any losses, injuries, or damages from the display or use of this information.

Connecting GPS using another Pro Mini and the I2C Bus

There are a few different ideas out there about how to connect GPS into your system. The problem is that the setup used with the mMWC shield uses the Pro Mini Arduino board and the Pro Mini has only one hardware serial port. This one port is needed for connecting to the board – either by FTDI or with the Bluetooth module. So this leaves the issue of how to connect the GPS to the board when the one serial port is already in use.

Alternatives Ways of Connecting GPS

When searching for solutions to this problem, instructions were found regarding the use of the SoftSerial library to turn a digital pin into another serial port. However, it was felt that this was not a good solution as the soft serial ports are not fast enough to read the GPS data. Another alternative was to add a switch which allows the user to select the GPS instead of the FTDI when in the air. Again this is not ideal, especially if Bluetooth is required to provide real time data when the aircraft is in flight and in range. The solution considered in this guide is to use the I2C bus. There is a piece of hardware that can be purchased which does all this for you but the board seems to only be available in the US or Hong Kong so it will mean a wait for delivery if it's going anywhere else in the world. The link provides details of how to buy one of these boards http://www.rctimer.com/product_762.html . Although the specific board is available to buy, the process below is very simple and quick to do, especially if you have a spare Pro Mini board lying around.

An Overview of the Process

A second Pro Mini Board will be connected into the system as a slave to the Arduino Pro Mini already used with the mMWC shield. The IMU is already using the I2C bus so the GPS will be assigned a separate address to enable communication from both devices to the master Arduino. Don't worry if you don't understand the terms slave and master or how the I2C bus works. If you follow the instructions outlined below you will get your GPS working with Multiwii very easily and very quickly.

Equipment Required

1. GPS – this tutorial will be using the U-BLOX NEO-6M GPS module and these are available from ebay for around £20.
2. A second Pro Mini board, 5V, the same as the one used with the mMWC shield, again it is available on ebay and will cost around £6.
3. The guide assumes that the mMWC shield board has been used to build the system detailed in Christian's mMWC-HowtoGuide V1.0.pdf and this work is adding on to the work he has already achieved.
<http://www.rcgroups.com/forums/attachment.php?attachmentid=5066631>

The instructions are split into 4 sections, the setting up of the GPS receiver for use with Multiwii, setting up the hardware, setting up the software and then testing out the whole system.

Instructions

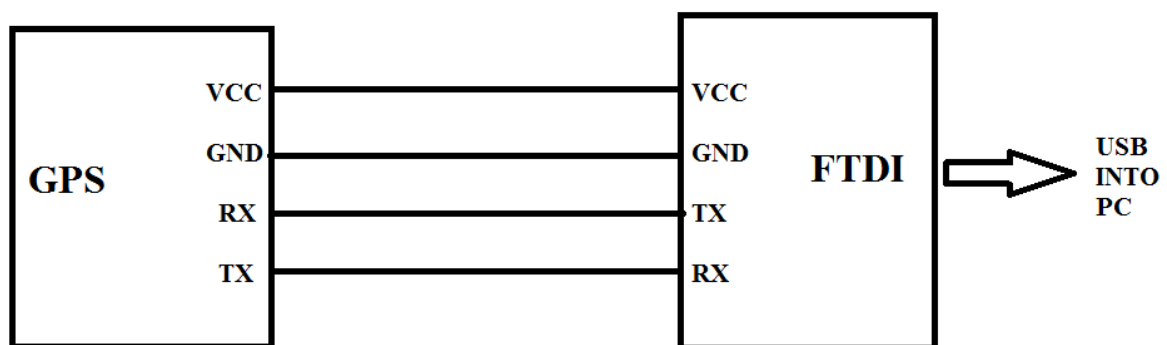
Configuring the GPS Receiver for Multiwii

The first thing that is needed is to change the settings in the actual GPS receiver to correspond with what Multiwii is looking for. There are 3 settings that need to be changed:

- Baud rate changed to 115200
- Update rate changed to 10Hz
- Output GGA, GSA and RMC frames (NMEA data)

These changes are going to be made using a configuration tool from U-Blox. So first of all download the U-Center from <http://www.u-blox.com/en/evaluation-tools-a-software/u-center/u-center.html>

Next take the FTDI cable and connect it up to the GPS directly and then into the computer. The Arduino is not being used at all at this point. So connect the GPS up as below:

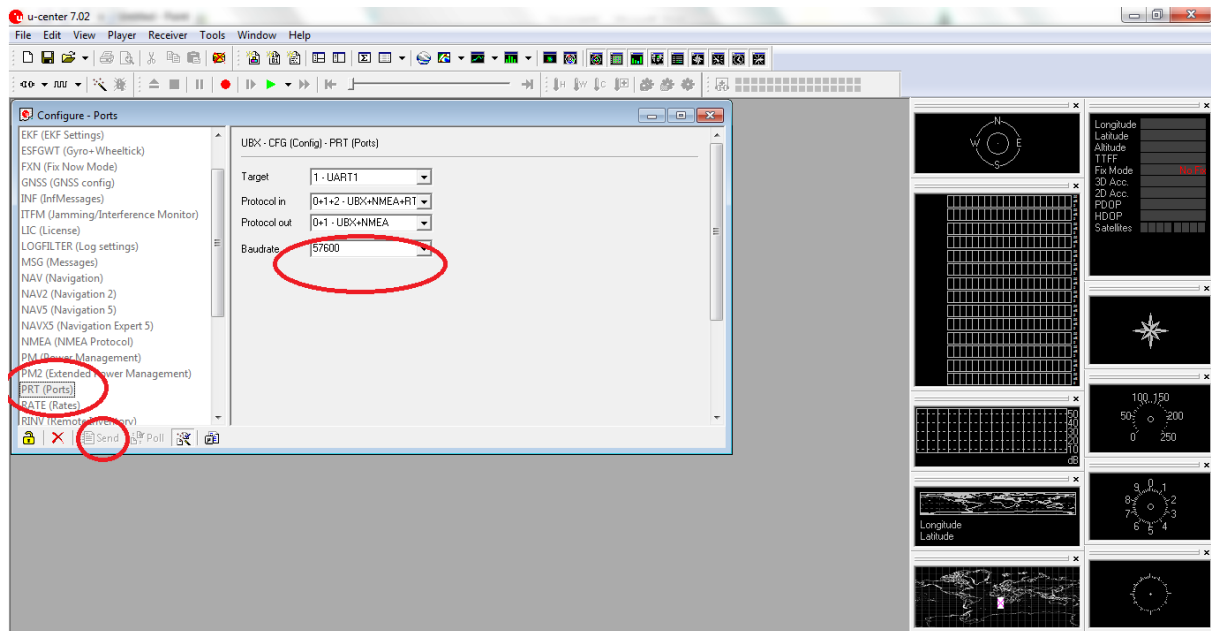


The pins on the GPS cable should be labelled but just make sure the TX on the GPS is going to the RX on the FTDI and vice versa.

When the FTDI cable is plugged into the PC, a solid blue light will be seen on the GPS. When this light goes from solid to flashing, it means the GPS is connected to the satellites and is receiving data. This can take a good 5 minutes if you are indoors! However that is not relevant at this stage of the process.

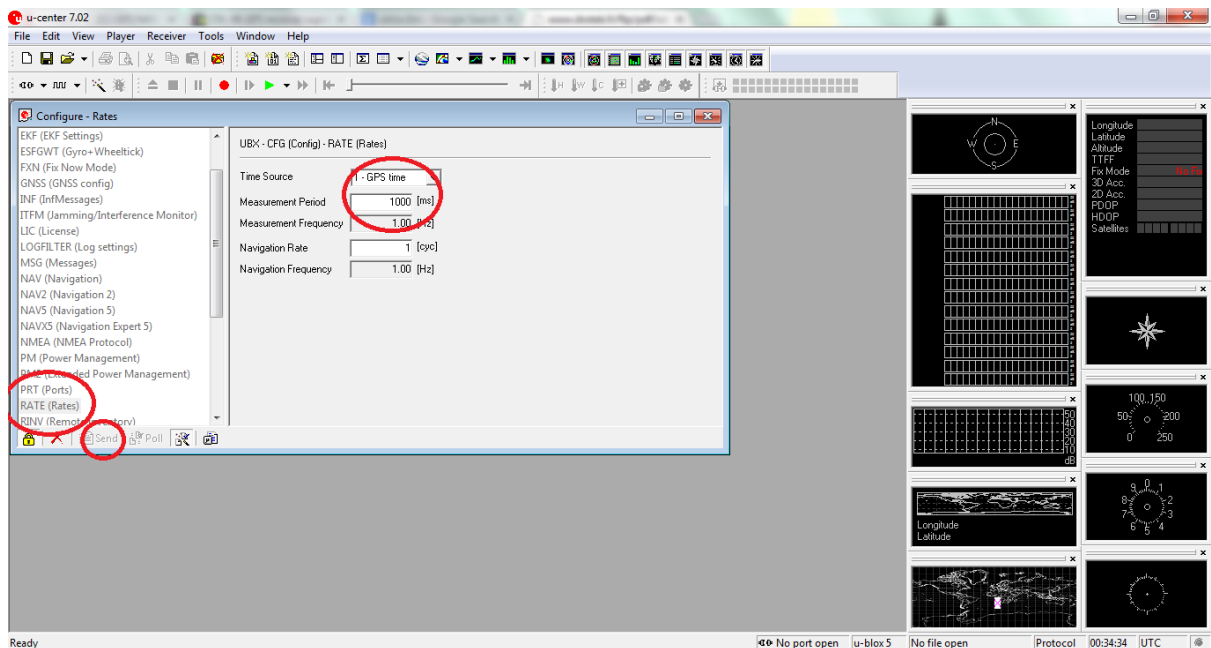
Open the 'U-Center'. On the top menu bar click receiver and choose the COMM port that the USB is plugged into. If you don't know which COMM port your FTDI cable is on, then click Start, right click on Computer and choose properties. Choose Device Manager on the left, click the arrow next to Ports and look to see which number COMM port your USB device is connected through.

Press Ctrl-F9 and this will open the configuration menu. Scroll down on the left and choose PRT as indicated below:

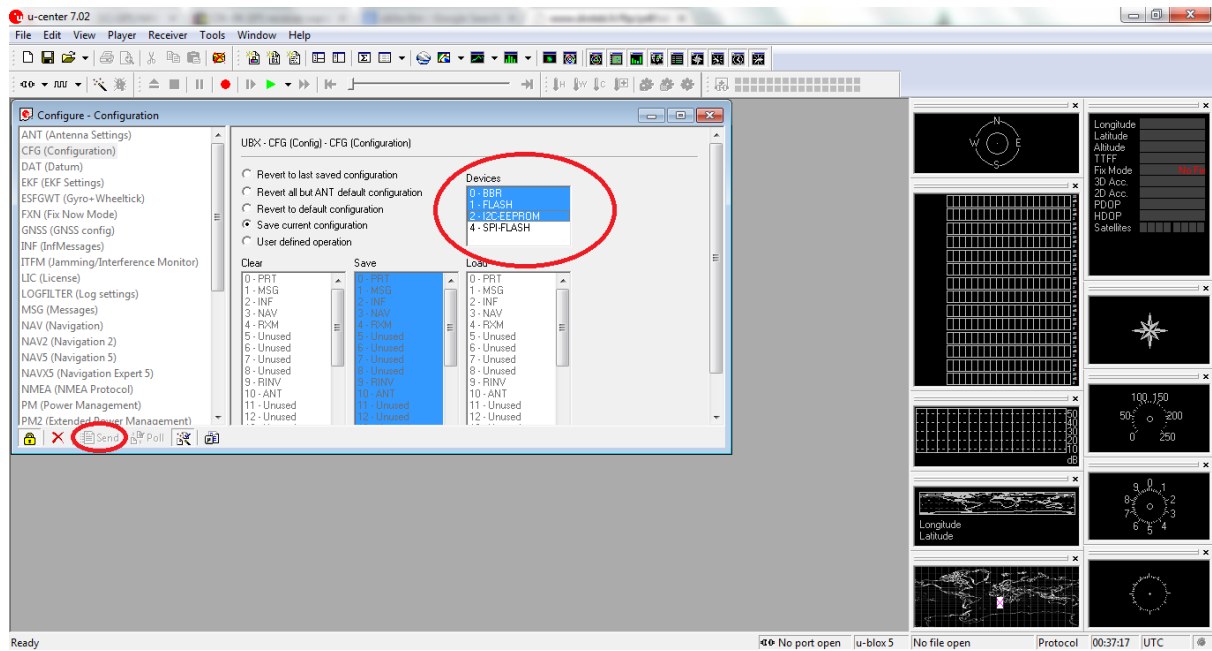


Choose PRT from the menu on the left, change the baud rate to 115200 and click Send. This will send the changes to the GPS. While in this menu the 'Protocol Out' needs changed to '1 – NMEA', so choose this and hit send again.

Now navigate to the Rate menu on the left. The Measurement Frequency needs to equal 10Hz and this is done by changing the Measurement Period to 100ms. Again press send to write it to the GPS.



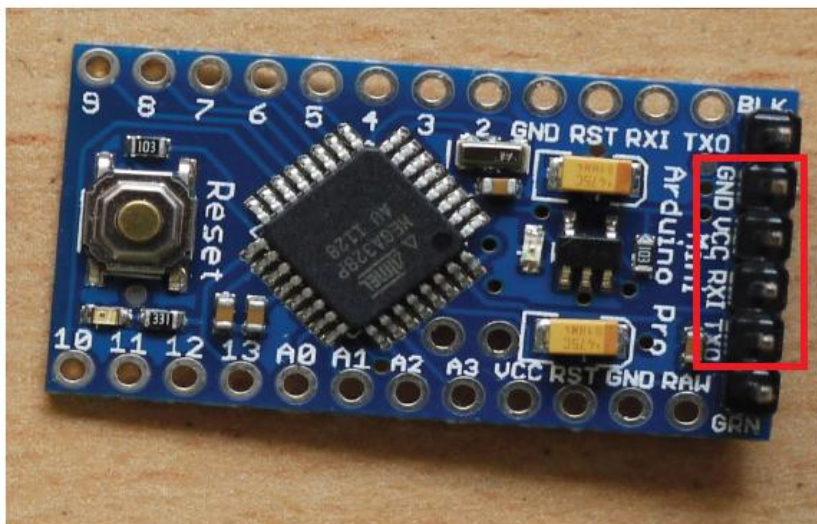
Next click on the CFG menu and it can be seen that '0 – BBR' and '1 – Flash' are highlighted in blue. In order for the settings to be saved on the device permanently, i.e. settings won't be lost when the GPS powers down, '2 - I2C-EEPROM' also needs to be highlighted. So highlight all 3 in blue, like below and then click send.



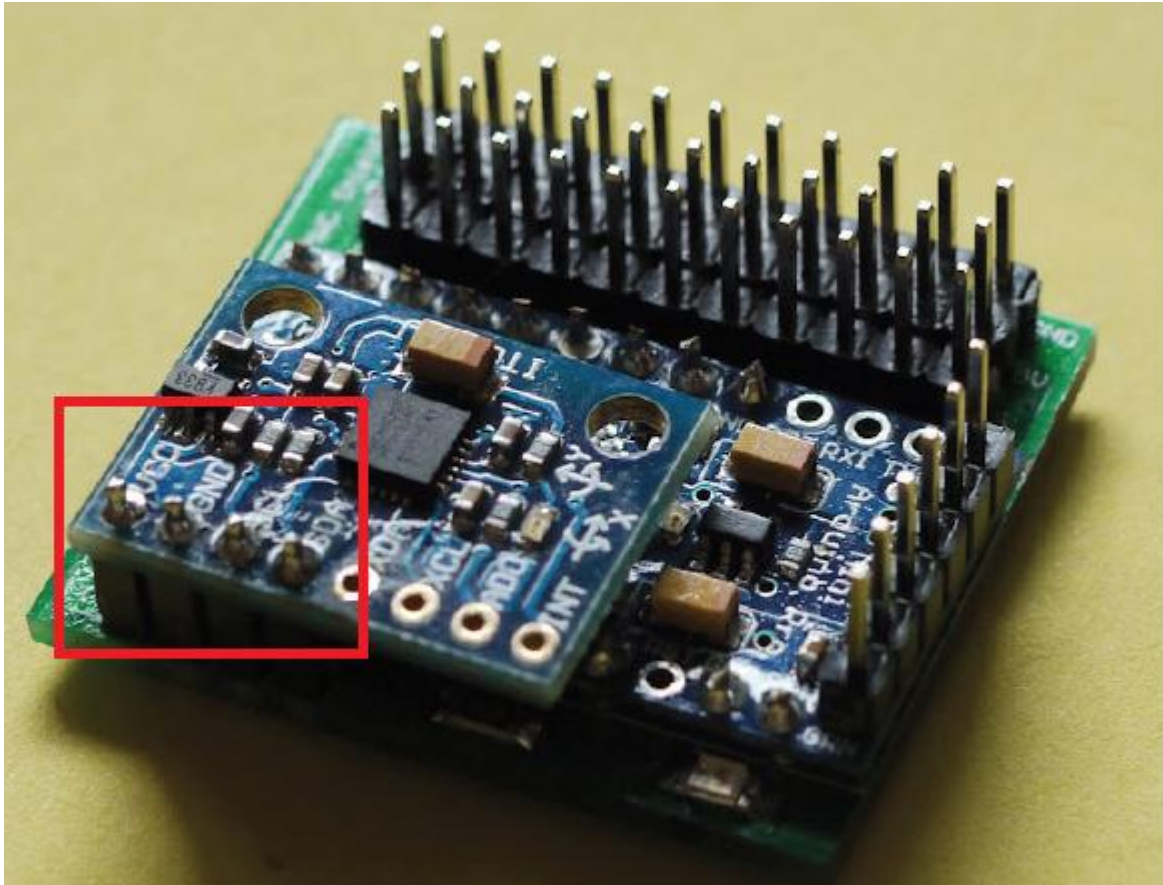
Now the GPS should be configured correctly for Multiwii and the FTDI cable can now be unplugged from the PC.

Connecting the Hardware

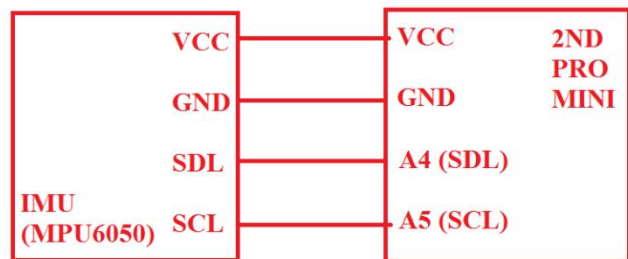
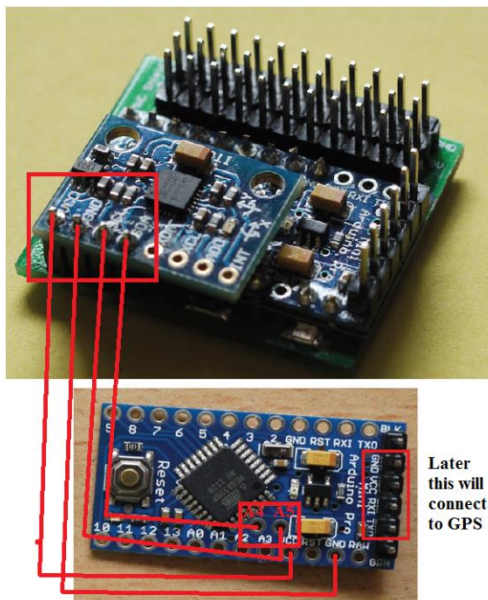
First of all solder some pins onto the second 'slave' pro mini so the GPS can easily be plugged in and out. As below in the picture, the 4 pins needed for the GPS connection are GND, VCC, RXI and TXO.



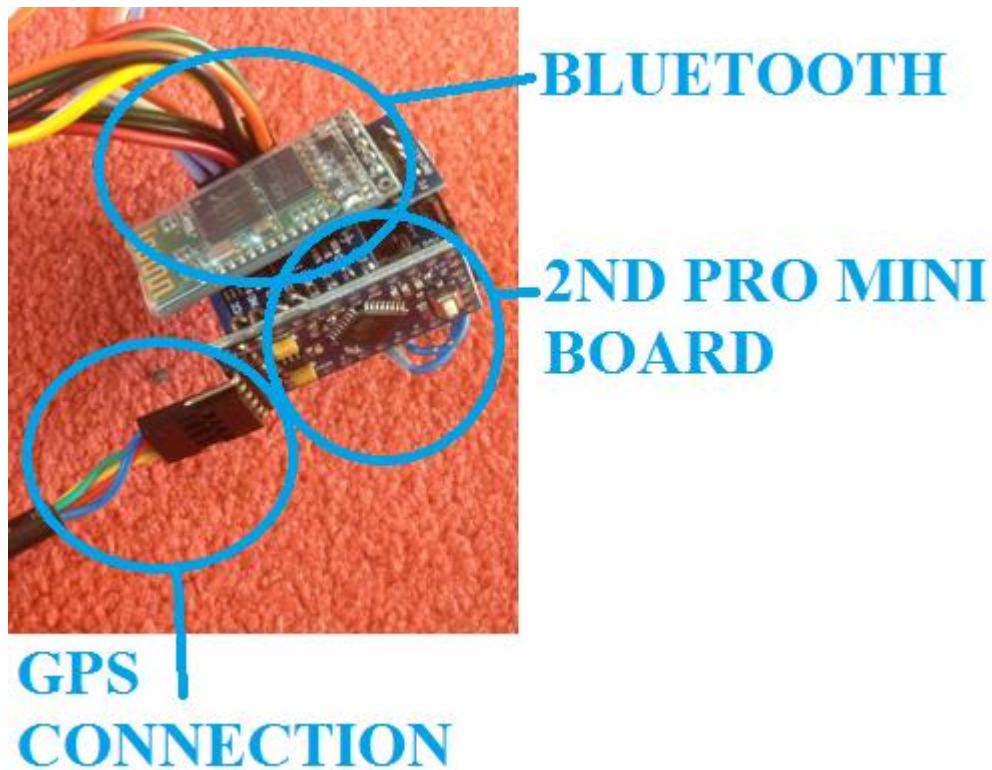
If you have followed Christian's How to Guide for the mMWC Shield, then you should already have a very neat and compact piece of electronics. The mMWC shield should be on the bottom with the Pro Mini sandwiched in the middle and the IMU on the top, as in the picture below.



The four pins highlight in red above are the pins which are going to be used to connect the 2nd Pro Mini board. See the specific connections in the picture below:



I mounted my 2nd pro mini at a right angle with the IMU, keeping the whole thing quite compact. See picture below:



That is the hardware side of things complete. Once those connections have been attached, the GPS can be connected up to the pins that were soldered onto the 2nd pro mini board earlier.

Setting Up the Software

First of all, disconnect the GPS unit from the 2nd pro mini board. Instead connect the FTDI cable into the 2nd pro mini board. Then download the following code:

https://code.google.com/p/i2c-gps-nav/downloads/detail?name=I2C_GPS_NAV_v2.2Beta1-r62.rar&can=2&q=

Extract the contents of the folder and open the file 'I2C_GPS_NAV_v2_2' in the Arduino IDE.

Just a few alterations are needed in the code before it can be uploaded to the second pro mini board. Move to the tab named config.h. Find the line:

```
#define I2C_ADDRESS 0x20
```

And change the address to 0x40, so the line becomes:

```
#define I2C_ADDRESS 0x40
```

Next find:

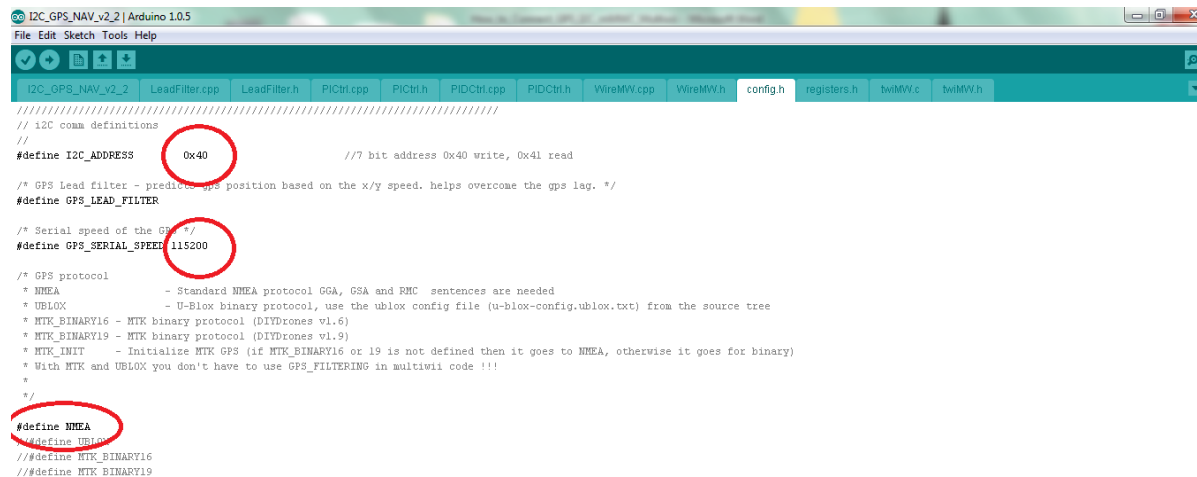
```
#define GPS_SERIAL_SPEED 115200
```

Make sure the speed is 115200, if it isn't then change it.

Lastly uncomment the line:

```
#define NMEA
```

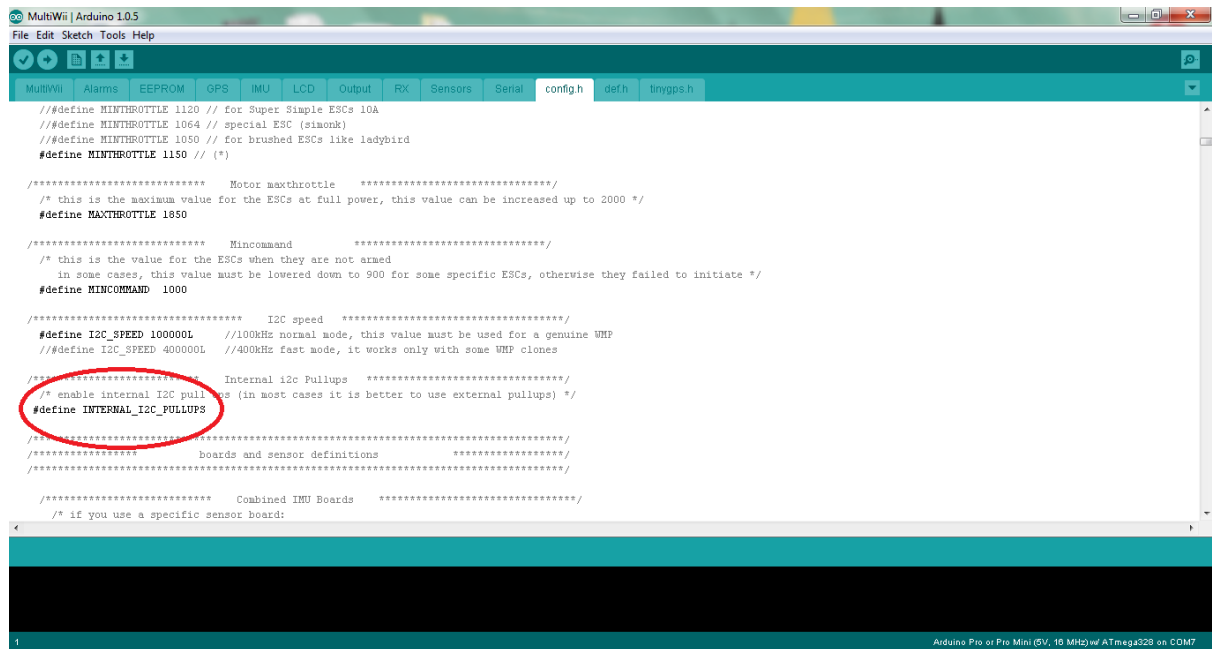
Make sure the #define UBLOX line is commented out. See the picture below:



Next save the changes. Make sure in the tools menu that the correct COMM port and board type has been selected. Then compile and upload the code to the board.

Disconnect the FTDI cable and connect the GPS back up in its place. Now connect the FTDI cable to the 1st board and open the Multiwii code.

Navigate to config.h and uncomment the following line about the I2C internal pull ups:



```
MultiWii | Arduino 1.0.5
File Edit Sketch Tools Help

MultiWii | Alarms | EEPROM | GPS | IMU | LCD | Output | RX | Sensors | Serial | config.h | def.h | tinygps.h

// #define MINTHROTTLE 1120 // for Super Simple ESCs 10A
// #define MINTHROTTLE 1064 // special ESC (simaonk)
// #define MINTHROTTLE 1050 // for brushed ESCs like ladybird
#define MINTHROTTLE 1150 // (*)

/***** Motor maxthrottle *****/
/* this is the maximum value for the ESCs at full power, this value can be increased up to 2000 */
#define MAXTHROTTLE 1850

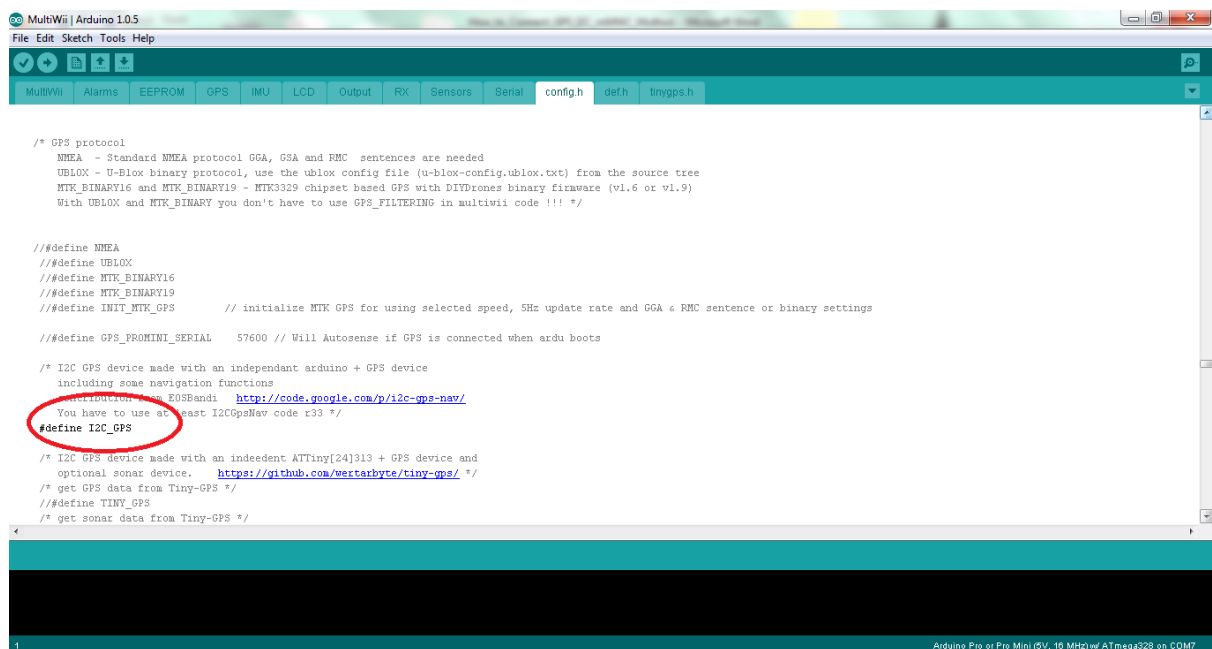
/***** Mincommand *****/
/* this is the value for the ESCs when they are not armed
in some cases, this value must be lowered down to 900 for some specific ESCs, otherwise they failed to initiate */
#define MINCOMMAND 1000

/***** I2C speed *****/
#define I2C_SPEED 100000L //100kHz normal mode, this value must be used for a genuine WMP
// #define I2C_SPEED 400000L //400kHz fast mode, it works only with some WMP clones

/***** Internal i2c Pullups *****/
/* enable internal I2C pullups (in most cases it is better to use external pullups) */
#define INTERNAL_I2C_PULLUPS

/***** boards and sensor definitions *****/
/***** Combined IMU Boards *****/
/* if you use a specific sensor board:
```

Next find the GPS section lower down and uncomment the line `#define I2C_GPS`. Make sure all the other lines above are commented out, right up to the beginning of the GPS section.



```
MultiWii | Arduino 1.0.5
File Edit Sketch Tools Help

MultiWii | Alarms | EEPROM | GPS | IMU | LCD | Output | RX | Sensors | Serial | config.h | def.h | tinygps.h

/* GPS protocol
NMEA - Standard NMEA protocol GGA, GSA and RMC sentences are needed
UBLOX - U-Blox binary protocol, use the ublox config file (u-blox-config.ublox.txt) from the source tree
MTK_BINARY16 and MTK_BINARY19 - MTK3329 chipset based GPS with DIYDrones binary firmware (v1.6 or v1.9)
With UBLOX and MTK_BINARY you don't have to use GPS_FILTERING in multiwii code !!! */

// #define NMEA
// #define UBLOX
// #define MTK_BINARY16
// #define MTK_BINARY19
// #define INIT_MTK_GPS // initialize MTK GPS for using selected speed, 5Hz update rate and GGA & RMC sentence or binary settings

// #define GPS_PROMINI_SERIAL 57600 // Will Autosense if GPS is connected when ardu boots

/* I2C GPS device made with an independant arduino + GPS device
including some navigation functions
contribution from EOSBandi http://code.google.com/p/i2c-gps-nav/
You have to use at least I2CGpsNav code r33 */
#define I2C_GPS

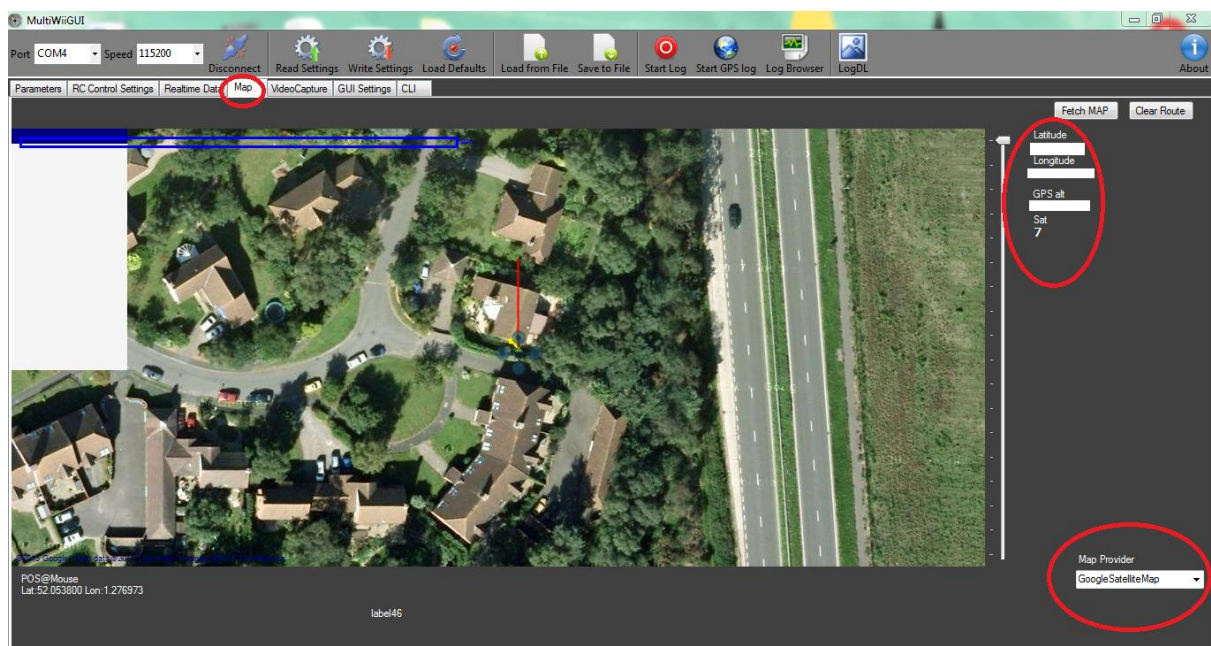
/* I2C GPS device made with an independant ATTiny[24]313 + GPS device and
optional sonar device. https://github.com/werctarbyte/tiny-gps/ */
/* get GPS data from Tiny-GPS */
// #define TINY_GPS
/* get sonar data from Tiny-GPS */
```

Next the I2C addresses will be checked to ensure they are correct for the GPS and the IMU. Navigate to the tab `def.h`.

Find the line below and check the address is `0x40`. If it's not, then change it to that value:



Choose the 'Realtime Data' tab. The number of satellites can be seen in the bottom left hand corner of the screen and this will be updated in real time. Now navigate to the MAP tab and the location of the GPS receiver should be shown on a map. The map provider can be chosen in the bottom right of the screen (google maps is being used below and works very nicely!).



Sources

The Drotek Manual was consulted to learn how to program the GPS receiver.

[1] Drotek, 'Manual - GPS Ublox NEO-6M', Version 1.0 Updated 2013 Mar 28, accessed 2013 August 05, available: http://www.drotek.fr/ftp/pdf/ublox_EN.pdf

The R-33 documentation provided some useful information regarding the settings needed on the GPS receiver in order for it to work with Multiwii. Also the documentation provided details of the flashing light sequences.

[2] Uploaded by Schaeffer A., 'Arduino based GPS and NAV co-processor with I2C protocol', uploaded 2012 May 23, accessed 2013 August 05, available: <https://code.google.com/p/i2c-gps-nav/downloads/detail?name=r33-documentation.pdf>

The I2C code was developed for a stand alone bit of hardware but it was used in this case to work with the Pro Mini without any real alterations.

[3] Uploaded by Schaeffer A., 'Arduino based GPS and NAV co-processor with I2C protocol', uploaded 2012 May 23, accessed 2013 August 04, available: https://code.google.com/p/i2c-gps-nav/downloads/detail?name=I2C_GPS_NAV_v2.2Beta1-r62.rar&can=2&q=

This document really is an add on to the work of Christian in the mMWC-How to Guide. Some pictures were used from the guide to show the GPS add on to the mMWC system.

[4] RC Group Forum., "mMWC-HowtoGuide V1.0.pdf -", accessed 2013 August 5, available: <http://www.rcgroups.com/forums/attachment.php?attachmentid=5066631>