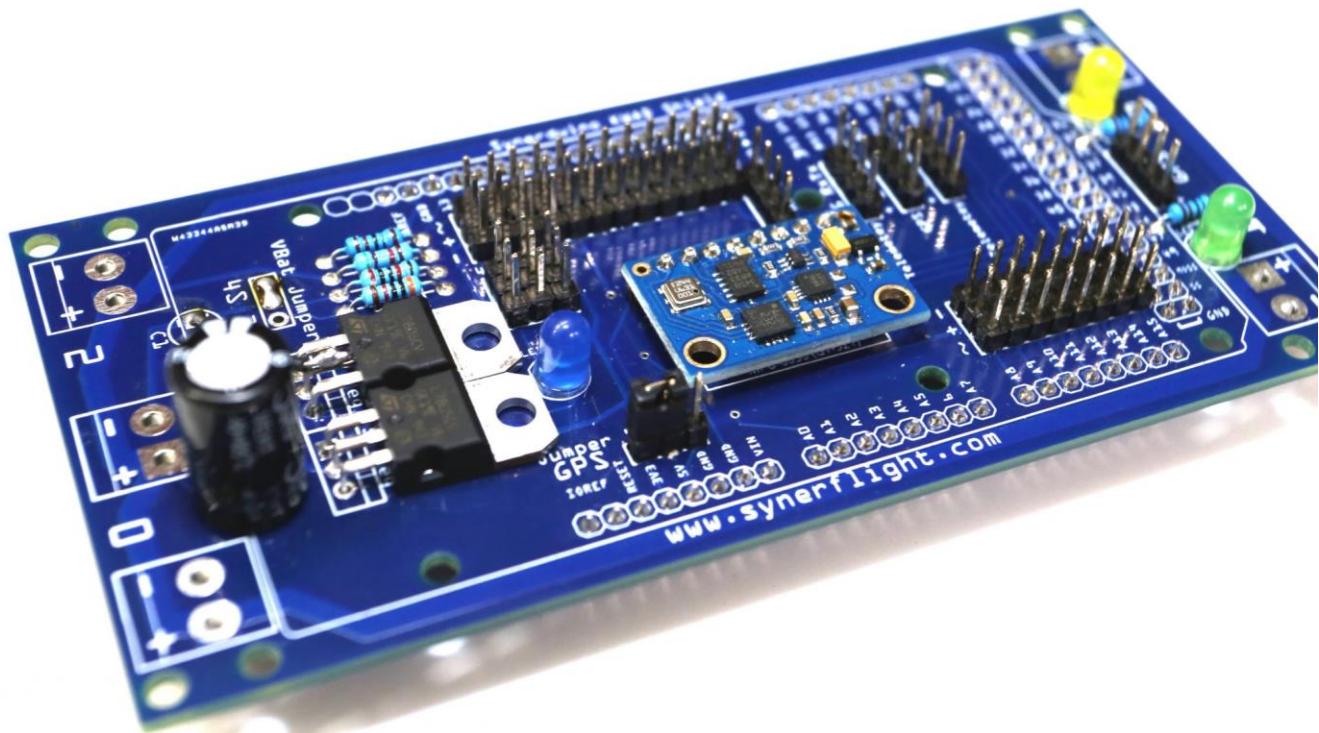


Synerduino Shield

Differential propulsion for Tanks and Boats



Hardware

- Synerduino shield
- Arduino2560 MEGA
- Tank or Boat Chassis Kit
- Brushed Motors 6V – 12V
- Gearbox (if your motor set came with it)
- Brush ESC for Car/Boat with Reverse function 6V-12V
- Ublox6 GPS
- Bluetooth

Tools needed

Screwdriver



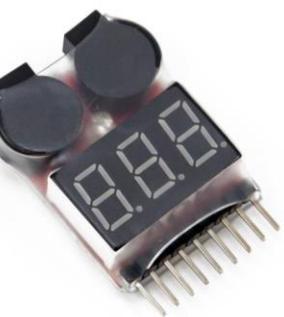
Cutter knife



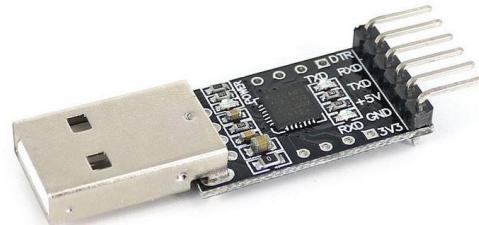
tape



Soldering Set



Voltage alarm



USB TTL FTDI

5A Balance Charger and Battery (3s 1100mah – 1300mah)



Wheel and Geared
motor



Car/Tank 2pcs

ESCs with Bi-
Directional function



2 Pcs of each

Boat Propeller with
Waterproof motor
Pods

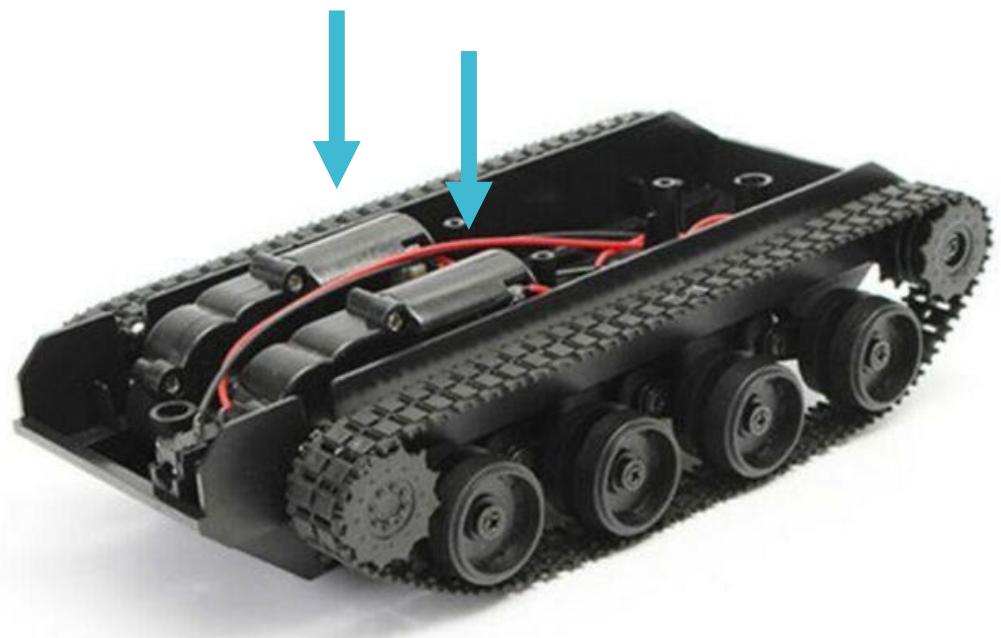


Boat 2pcs

Brushed or Brushless the BLDC motor should match your Electronic Speed controller

Differential Drive

2pcs Motor Differential drive



2pcs Motor Differential drive

Steering Drive

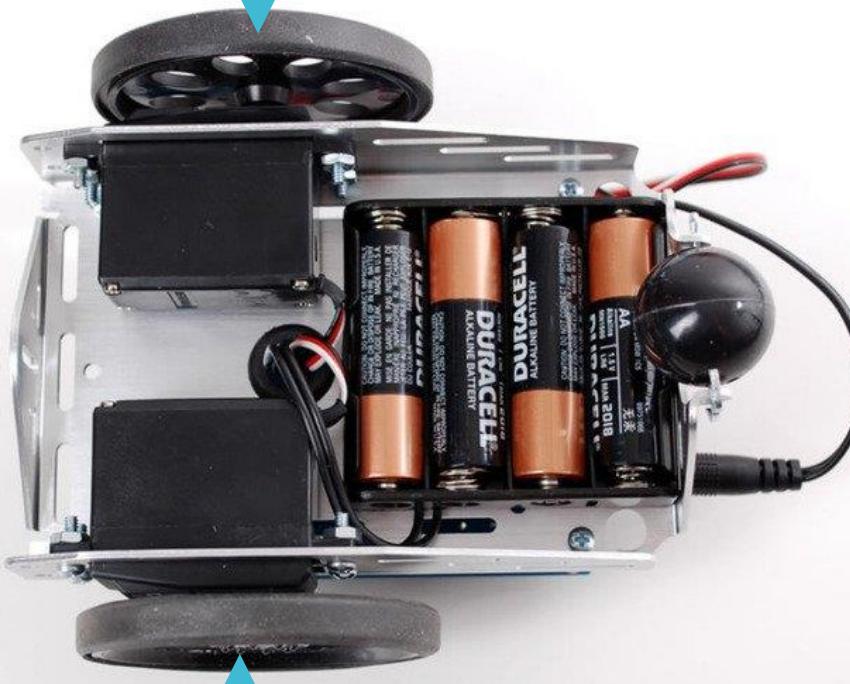
Motor & Steering Servo



Motor & Rudder Servo

Differential Drive with CR Servos

Continues Rotational Servo



Continues Rotational Servo



Servos Design to rotate 360Degrees with proportional speed and directional control



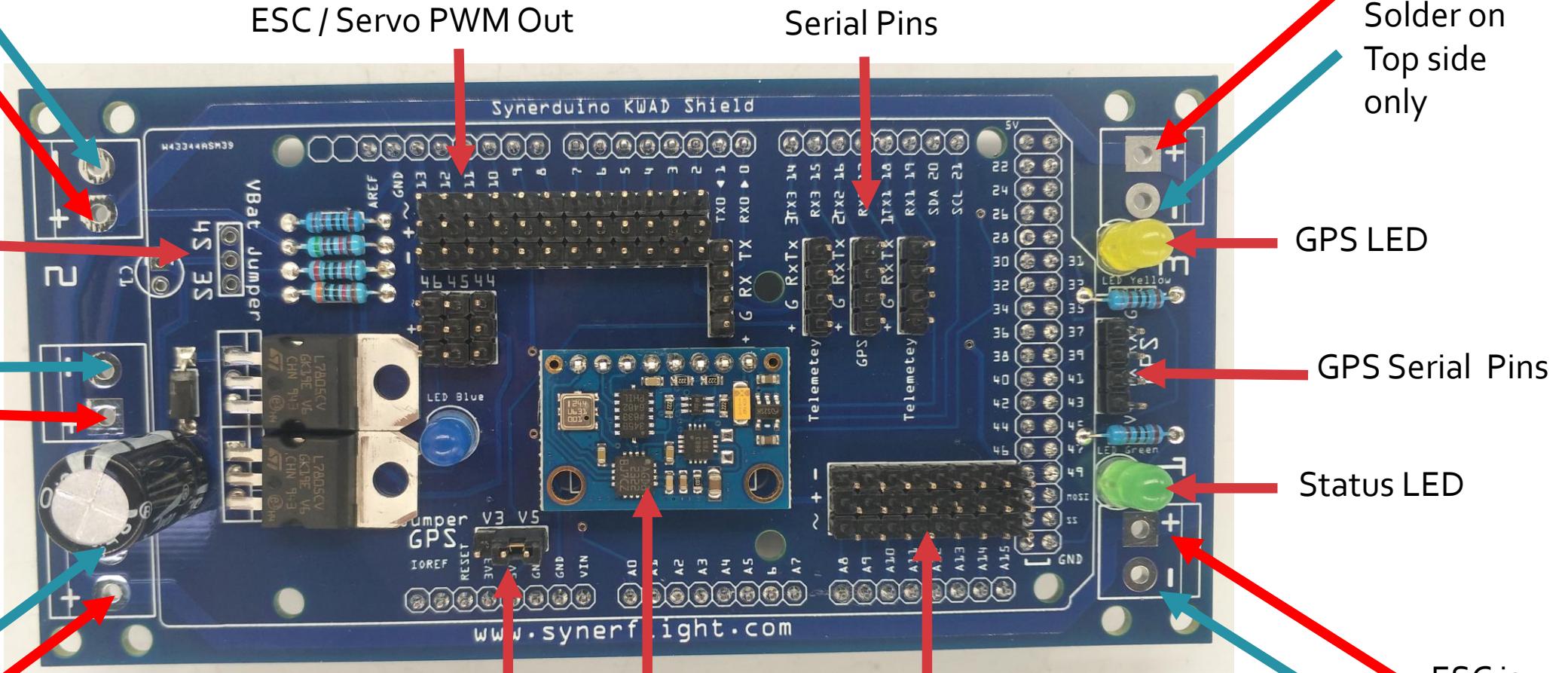
1000ms Reverse – 1500ms Neutral – 2000ms Forward

For 4WD configuration pair up two servos with a Servo Y splitter

Synerduino Kwad Shield Beta GY801

ESC is
Solder on
Top side
only

Note : surface mount your solder ESC wire make sure it doesn't penetrate to the bottom of the board

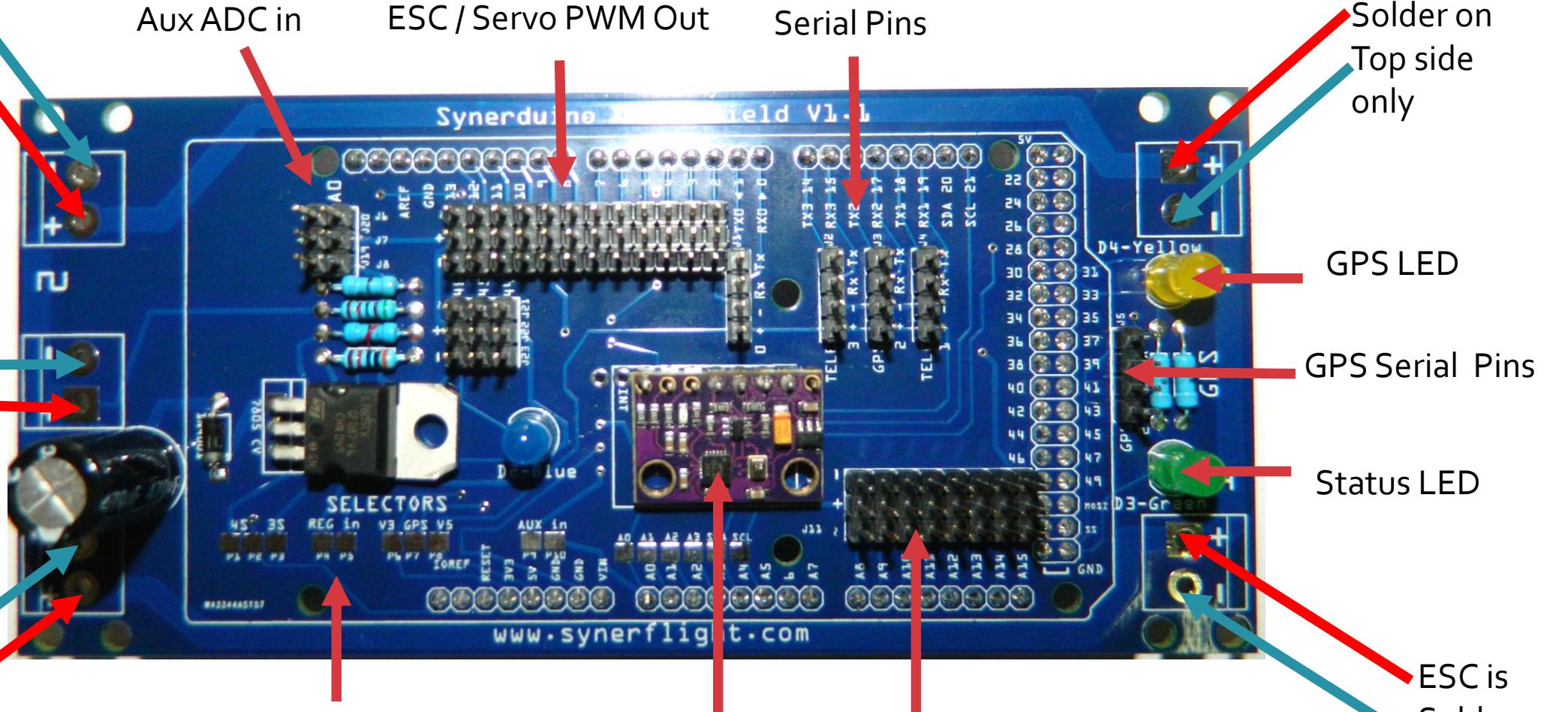


IMU : L3G4200D Gyro / ADXL345 Accelerometer / BMP180 – 85 Baro / MMC5883 Mag

Synerduino Kwad Shield V1.1 GY91

ESC is
Solder on
Top side
only

Note : surface mount your solder ESC wire make sure it doesn't penetrate to the bottom of the board

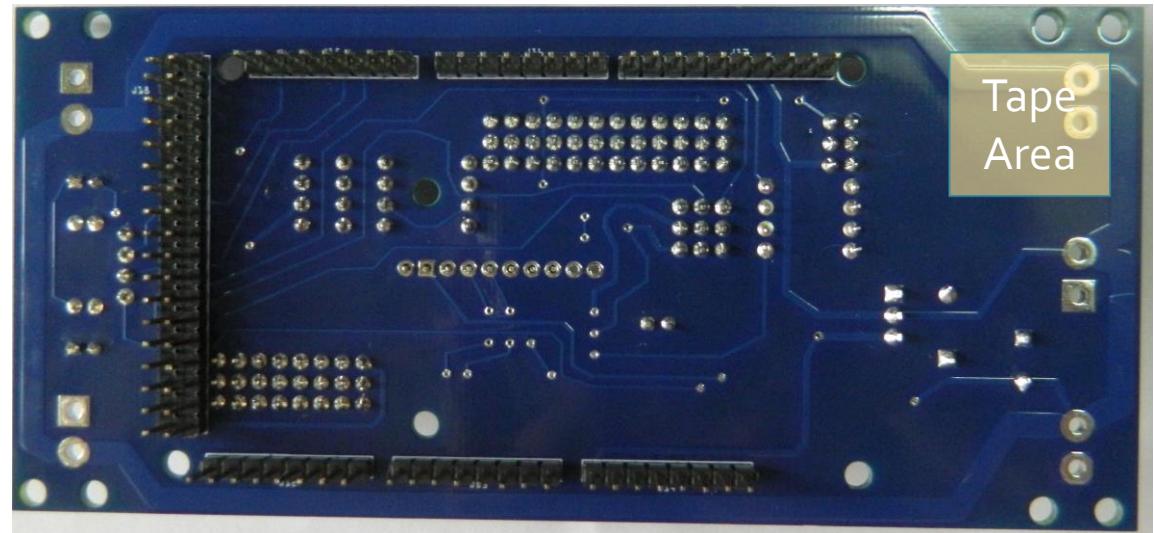
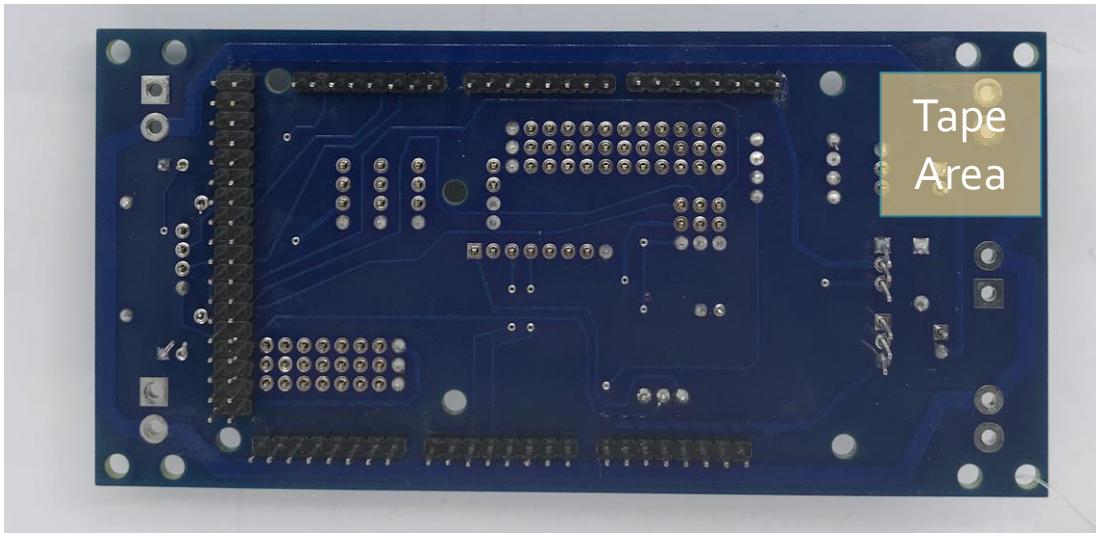


IMU : MPU-9250 & BMP280

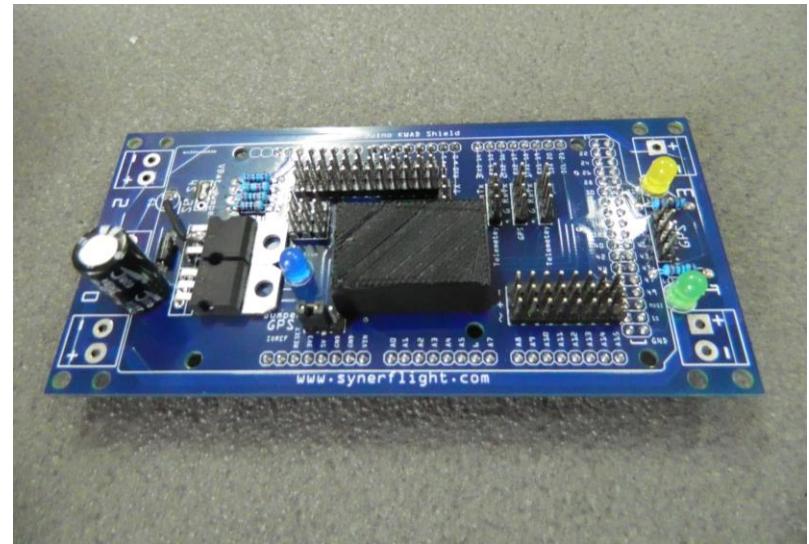
Synerduino Kwad Shield Preparation

Ensure insulation from the Arduino board add tape on these areas

BETA GY801 & V1.1 GY91



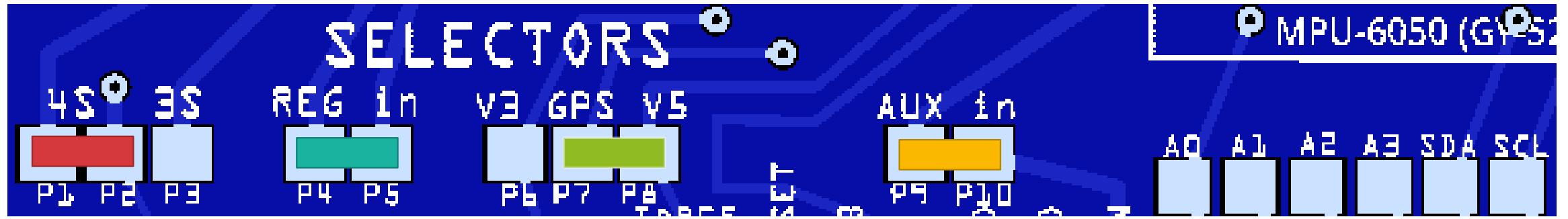
Use PVA to glue the cover to the IMU sensors seal the rim of the cover all the way



Synerduino Kwad Shield V1.1 (GY91)

Added the Selector Jumper Pads to the main board





Battery cell
monitoring
4s or 3s

5V Regulator
from battery

GPS Pins V+
voltage in front
of the board

AUX in

AO A1 A2 A3 SDA SCL

Analog 0-3 & I2C external
sensors

To use onboard battery monitoring with Aux in Set to 3s if your running 1s-3s battery / set to 4s if your running 4s battery / Leave it open when using Aux in as External sensors or using 5s to 6s

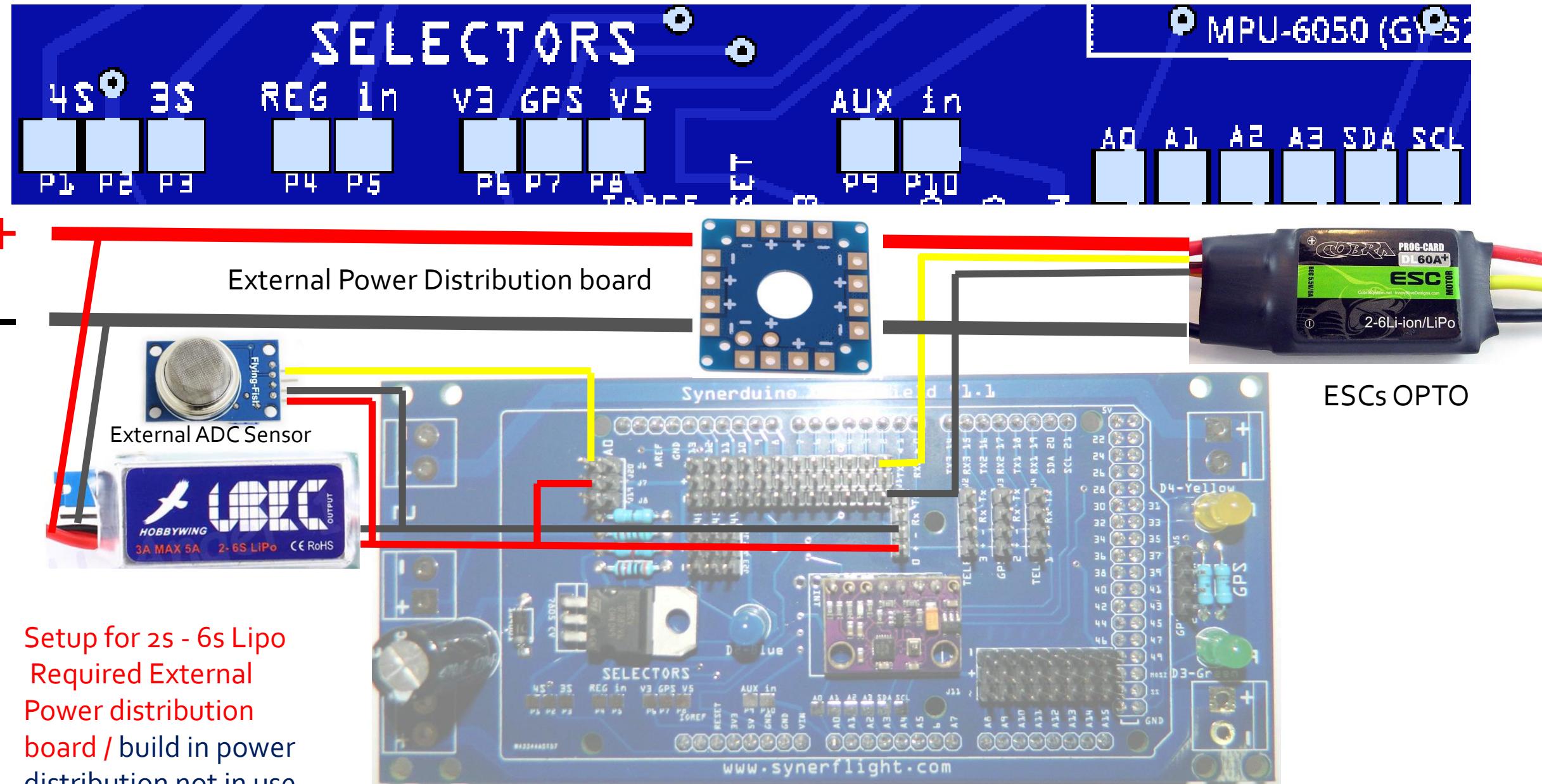
Reg In – short the Pads for using regulator to power and build in power distributor the synerduino and Arduino board

2nd GPS pin with voltage selector 5V for Regular GPS / 3V for external I2c sensor such as Magnetometer

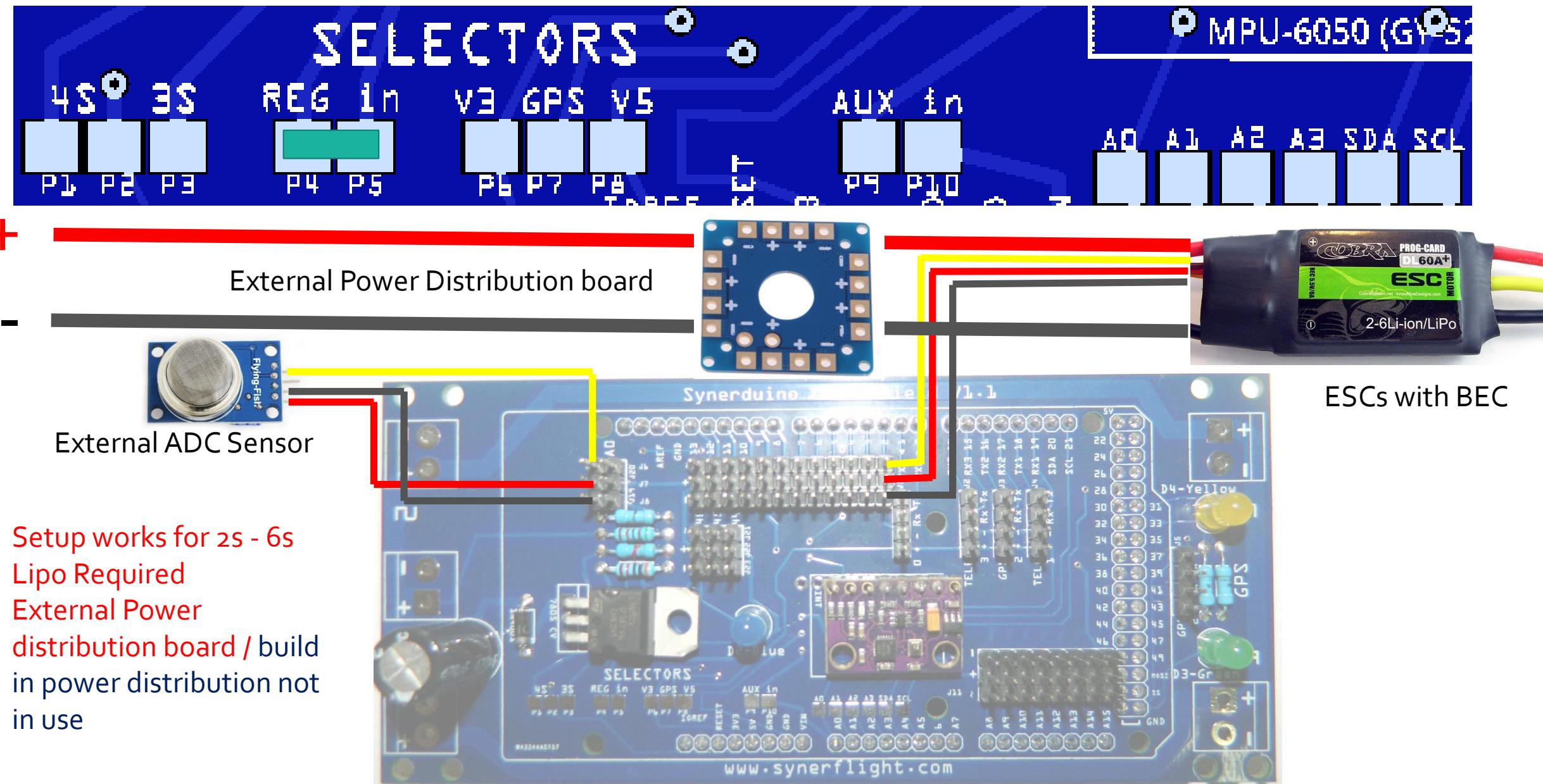
Aux in- leave it open for utilizing the Ao Pins for External ADC sensors / Short the Pads to use build in battery monitoring . Cell Selector must be set to 4s or 3s

SDA SLC - I2C input for external sensors such as GPS with build in Magnetometer

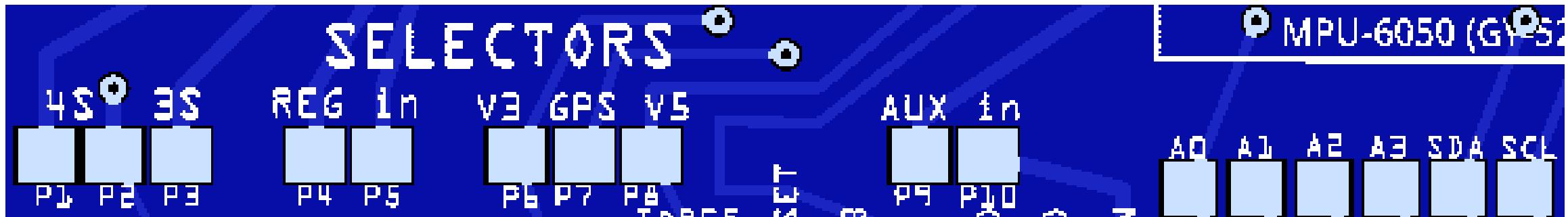
Default - Ao External ADC sensor , Require 5v UBEC applied to the 5V Serial pins & for OPTO ESC



Reg in only - Ao External ADC sensor , Require ESC with UBEC applied to the 5V PWM pins

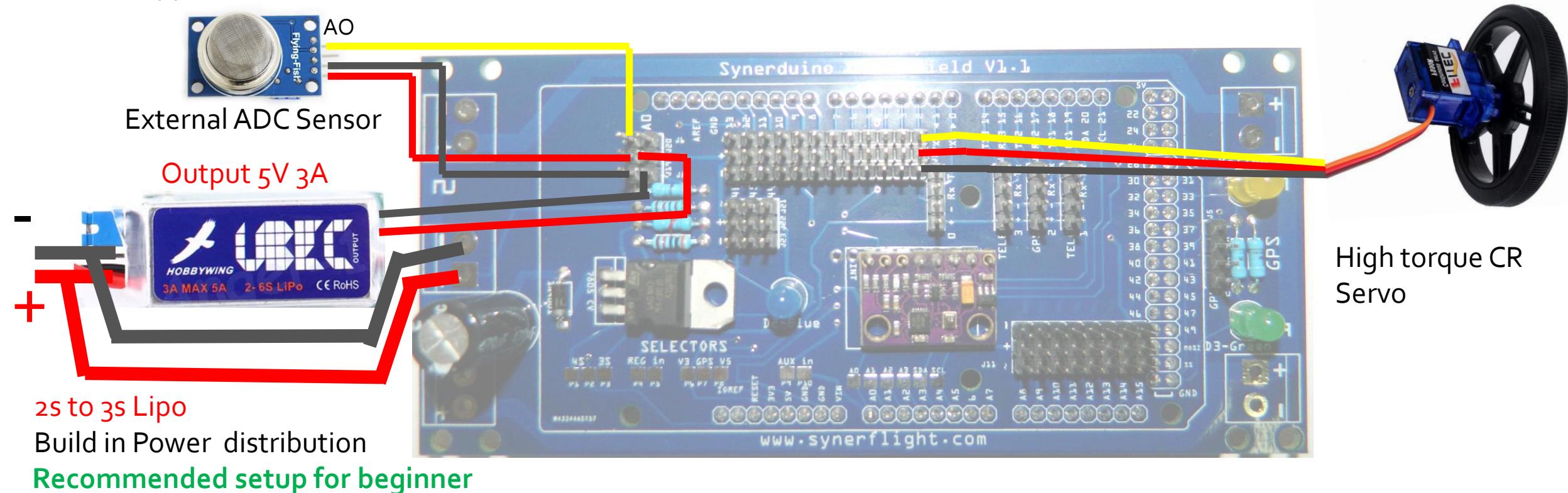


ISOLATED - Ao External ADC sensor , ESC BEC or OPTO applied to the 5V PWM pins

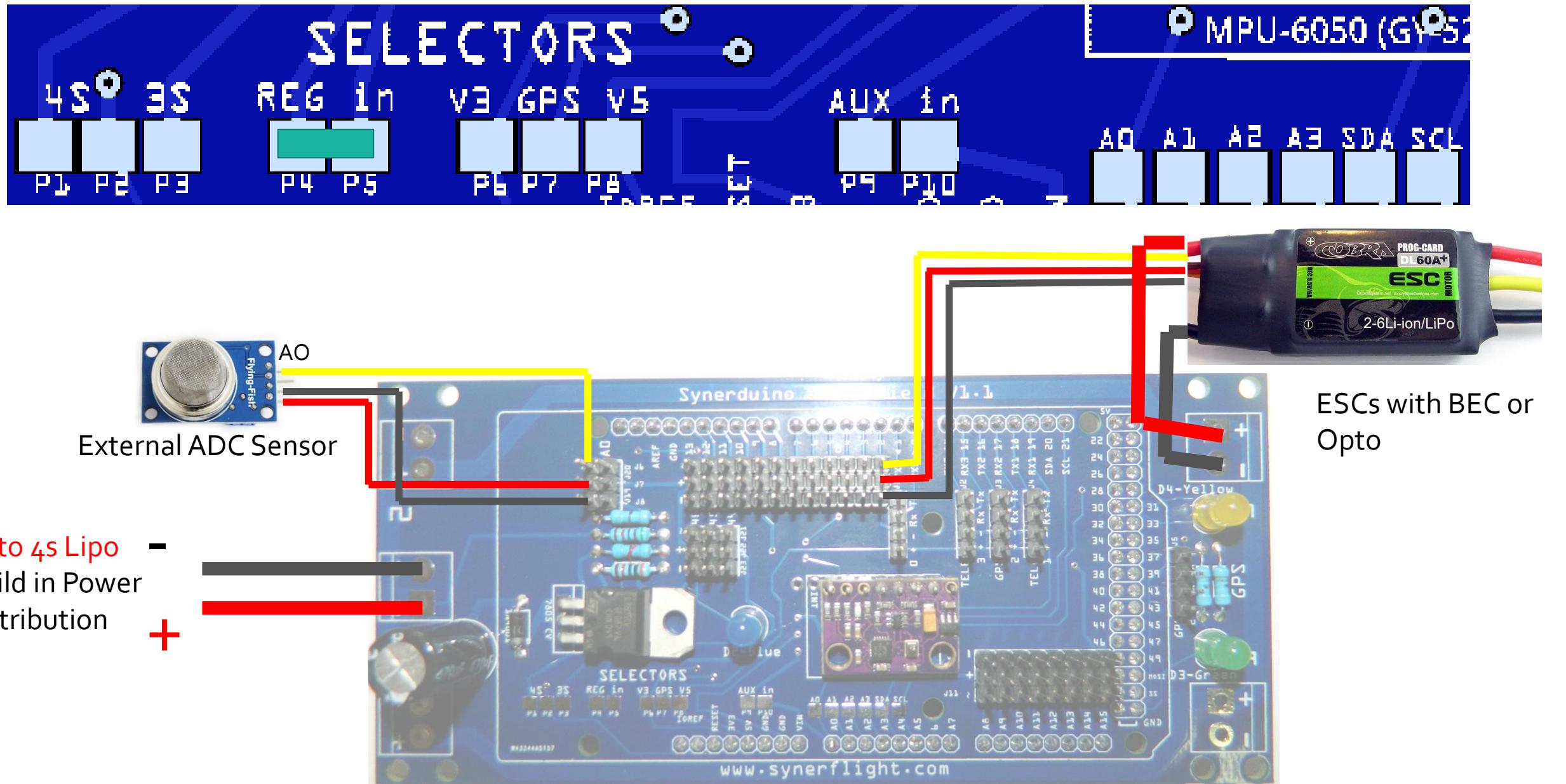


This setup is ideal for Servo or Sensor Intensive application that requires higher current driven across the power rails

Application such as Continues Rotational Servos and several ADC sensors

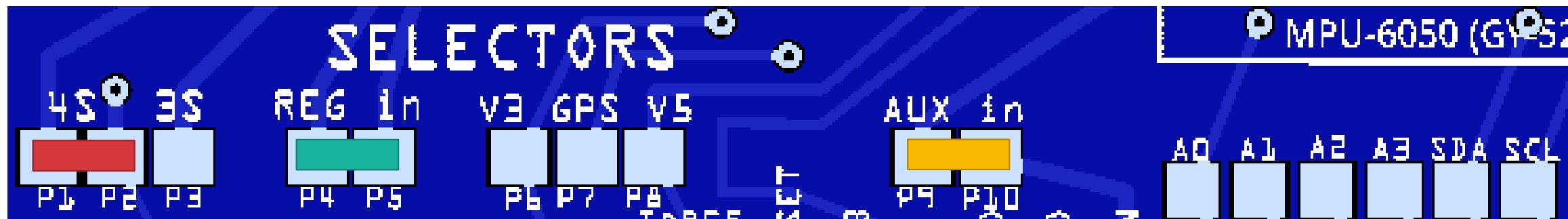


Reg in only - Ao External ADC sensor , ESC BEC or OPTO applied to the 5V PWM pins

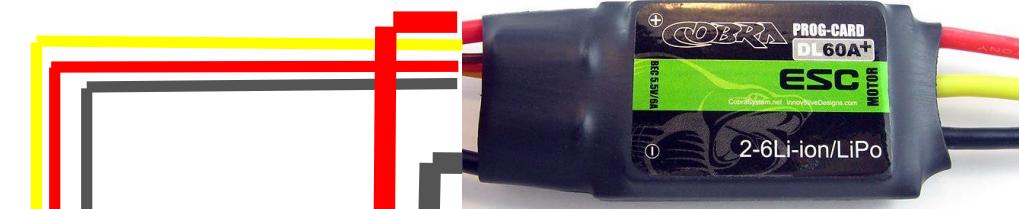


Recommended setup for beginner

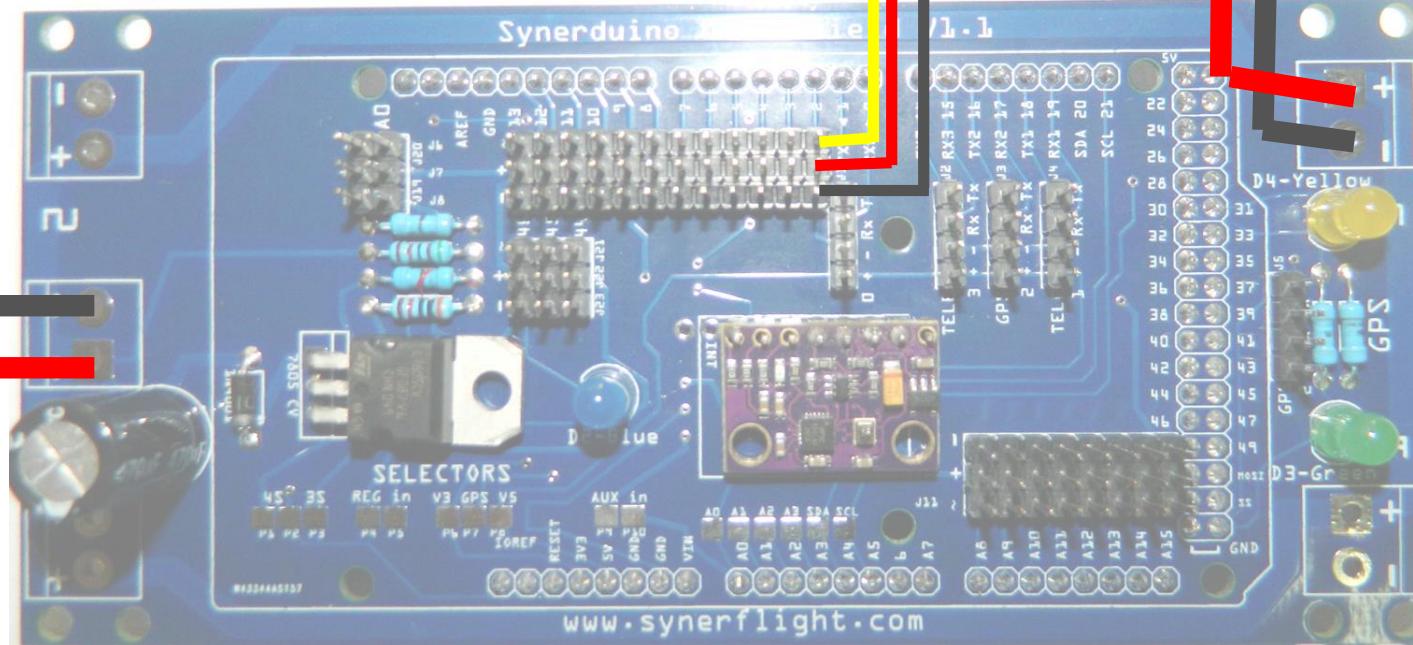
Reg & Vbat - Ao as Battery voltage monitor , ESC BEC or OPTO applied to the 5V PWM pins



For those who would use the build in battery monitoring circuit upto 4s lipo ensure the Cell Count and Aux in is jumped before powering up



ESCs with BEC or Opto



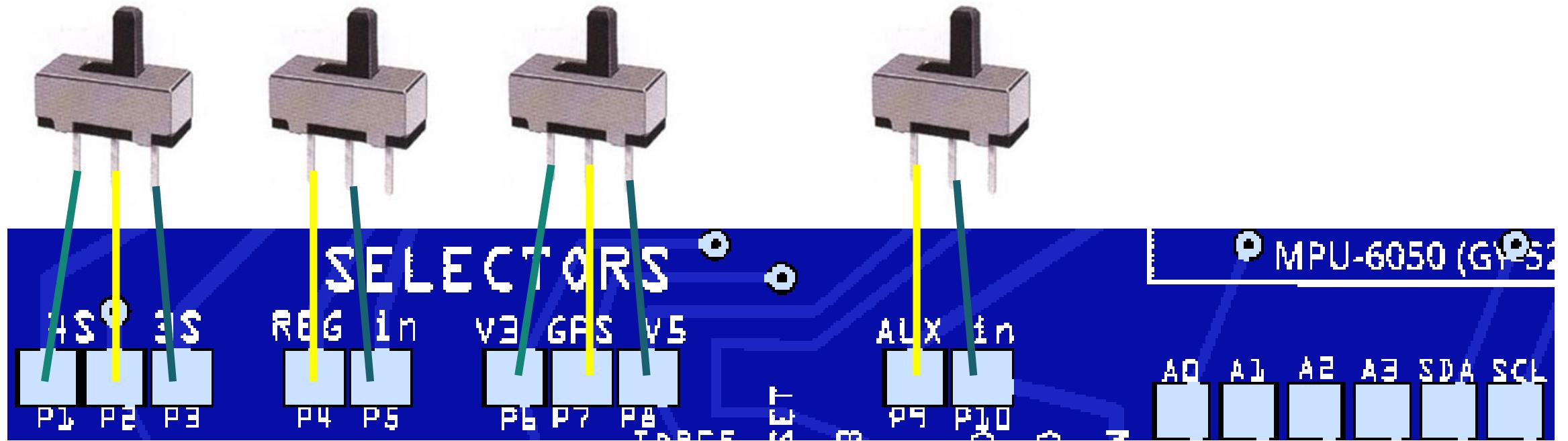
2s to 4s Lipo
Build in Power distribution

-



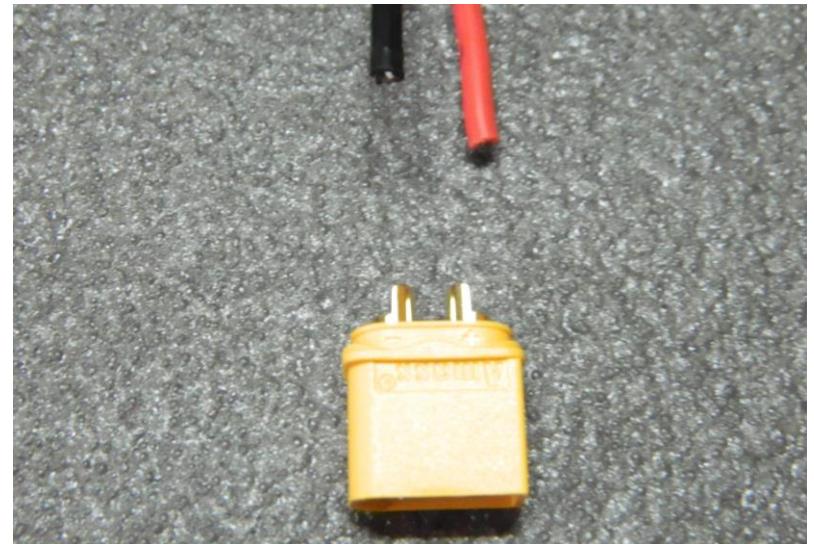
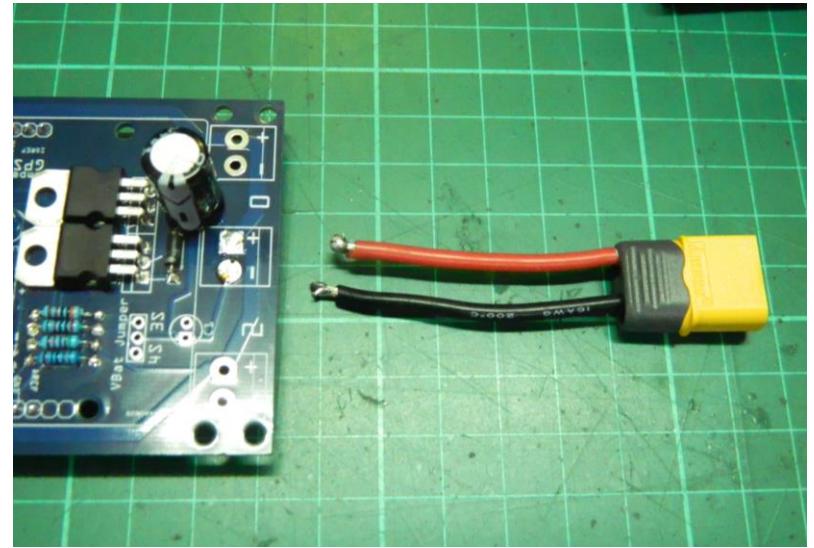
Recommended setup for beginner

For those who thinks they need to change setup from time to time

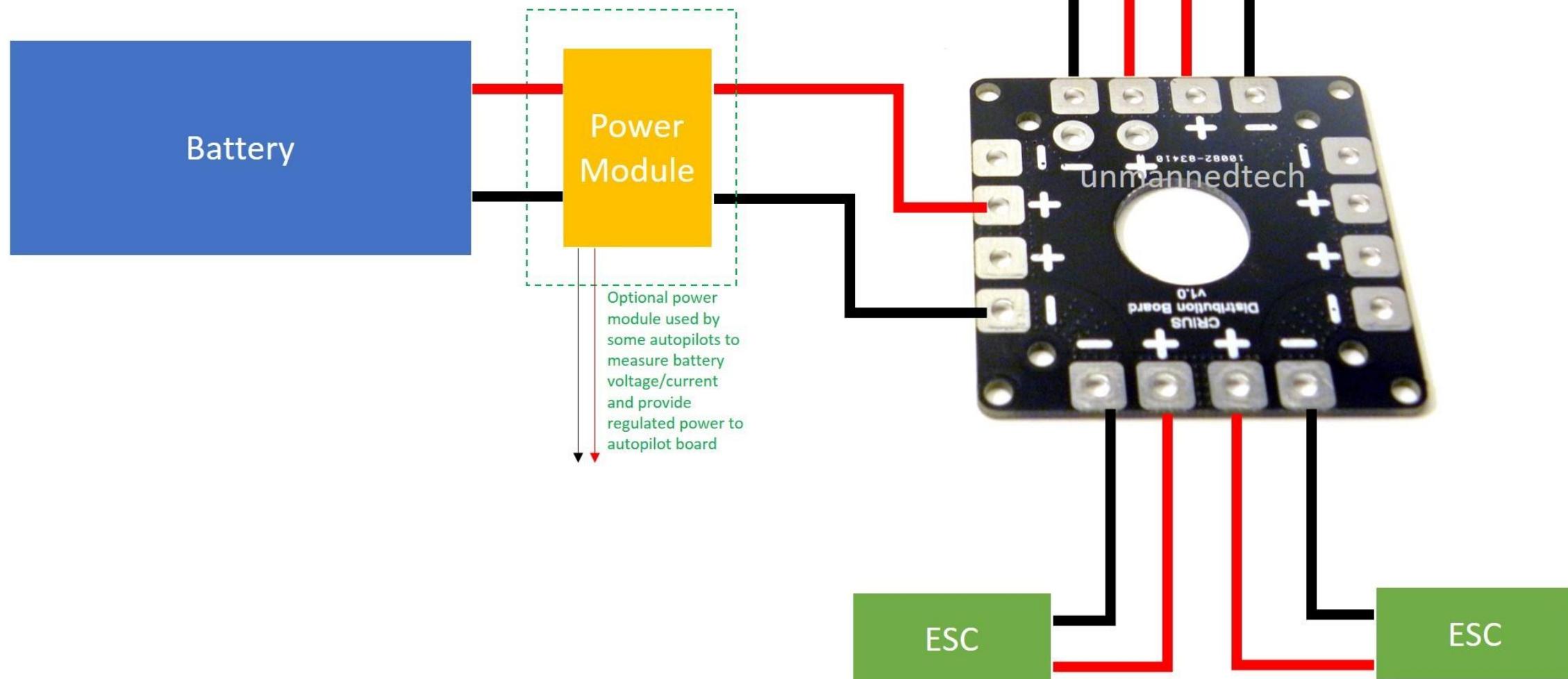


For those who would use the build in battery monitoring
circuit upto 4s lipo **ensure the Cell Count and Aux in is jumped
before powering up**

Pls check the polarity of the XT6oPlugs to match up with the board's polarity



For those running power higher than 4s Lipo advice to use a external power distribution board and UBEC



Arduino Board Preparation

Ensure insulation from the Arduino board add tape on these areas



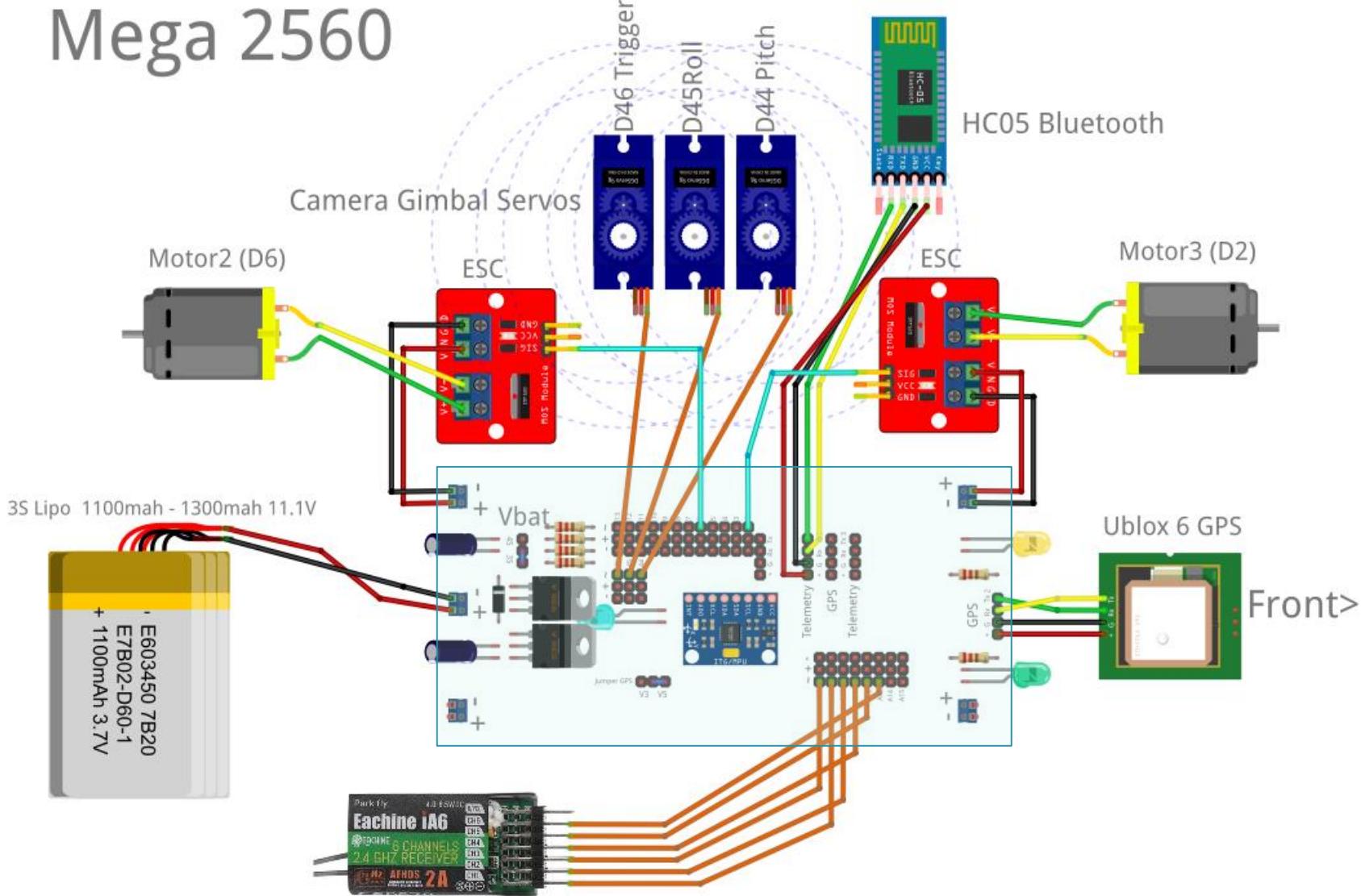
2560 MEGA



SMD UNO 328

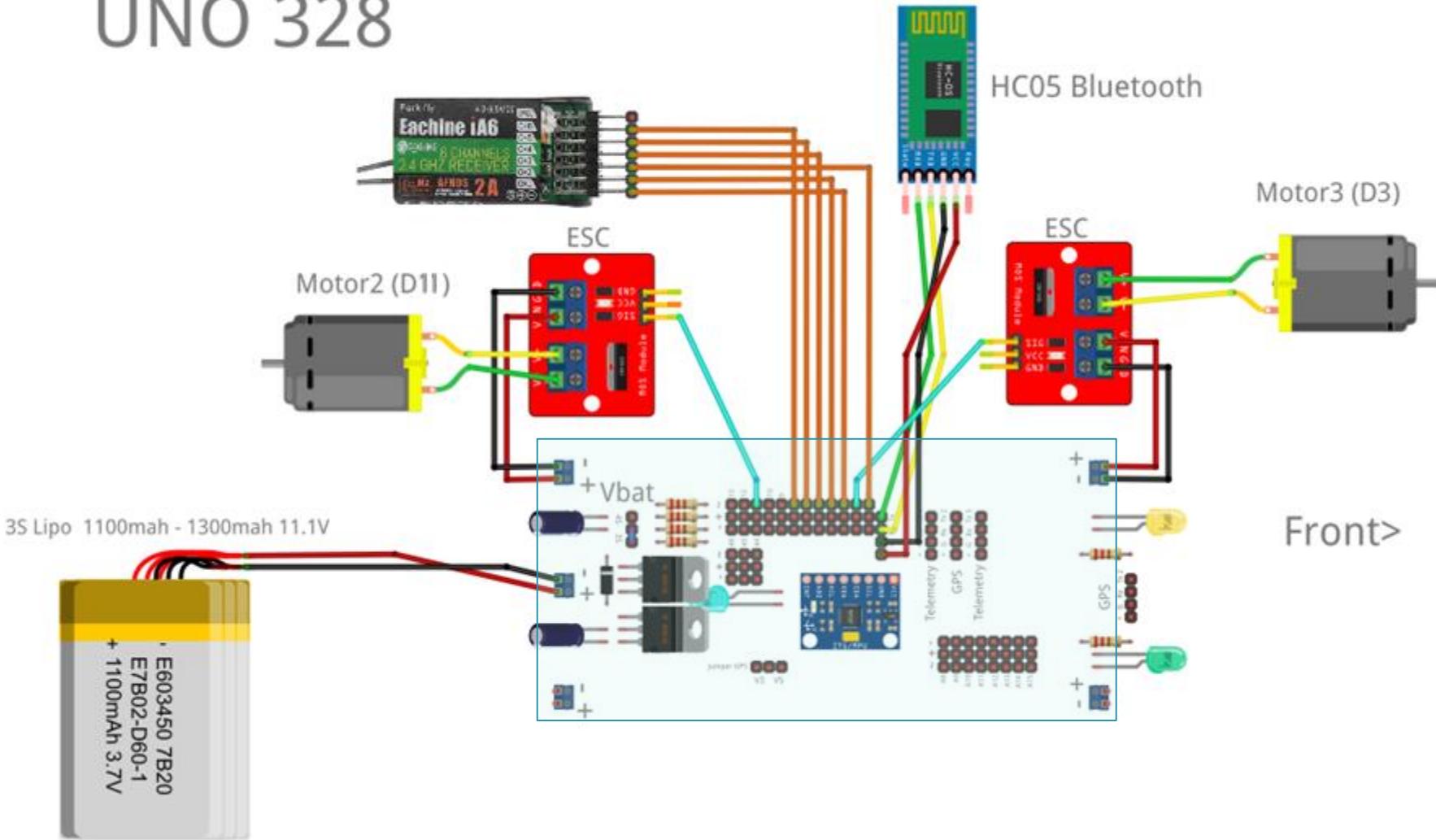
Differential Drive

Mega 2560



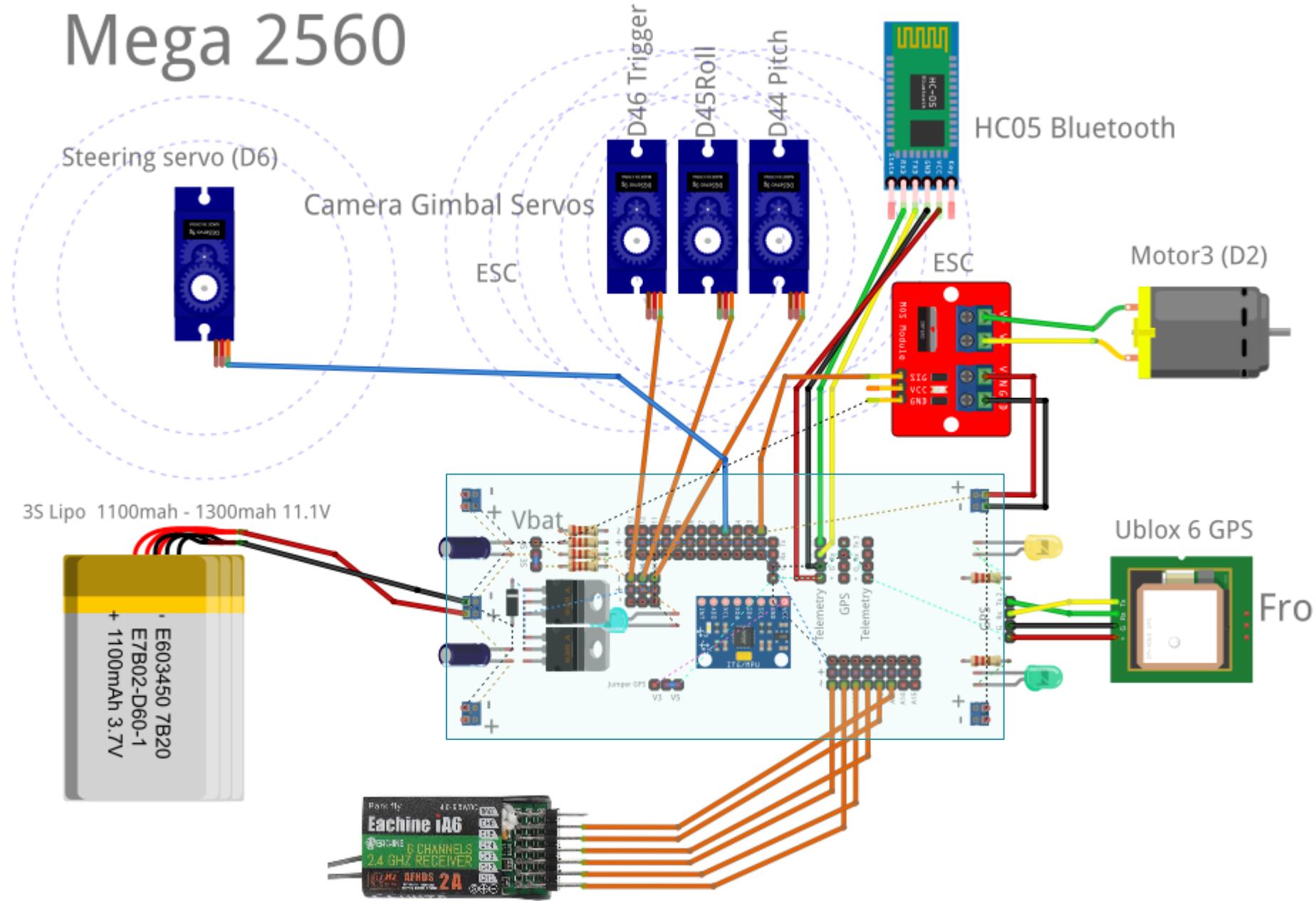
Drive Differential

UNO 328



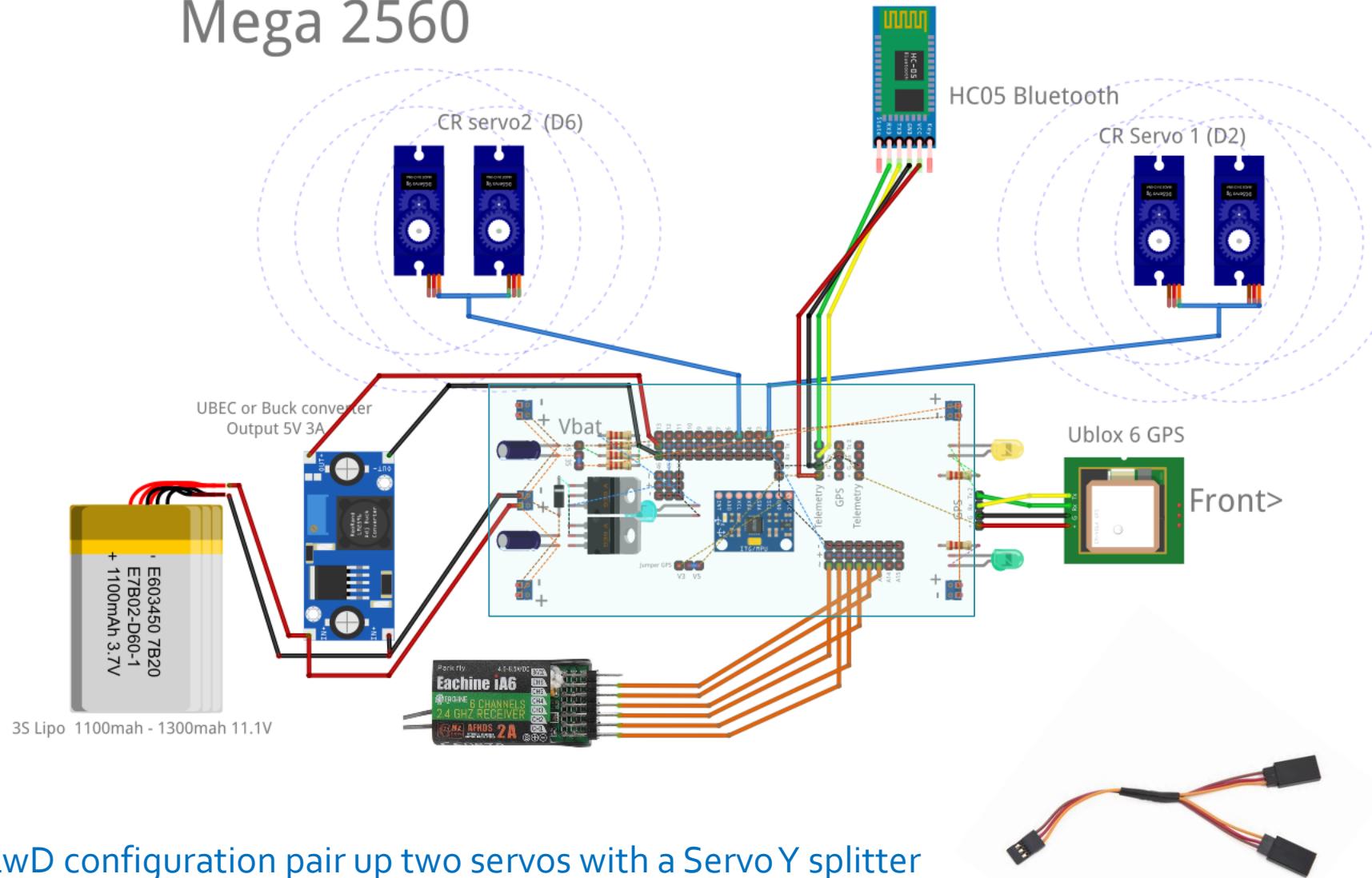
Steering Drive

Mega 2560



Conventional Differential Drive – using Continuous Rotation Servos

Mega 2560



For Tank and Boat chassis
Motors use Pins

Select (#Define Bi) Frame on
the CONFIG.H

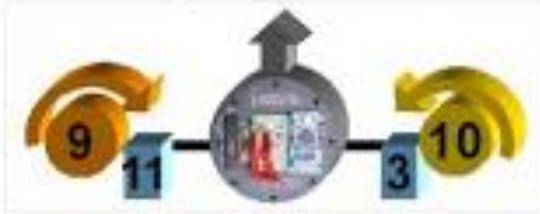
D11 & D3 UNO 328
MotorA & MotorB

D6 & D2 MEGA 2560
MotorA & MotorB

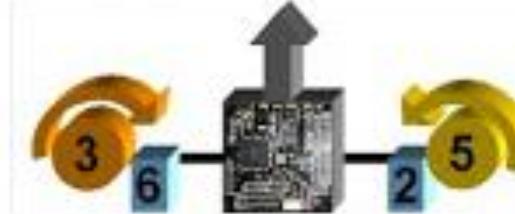
Conventional Car
D2 Motor Drive
D6 Steering Servo

BICOPTER avatar style

for Arduino 328p:



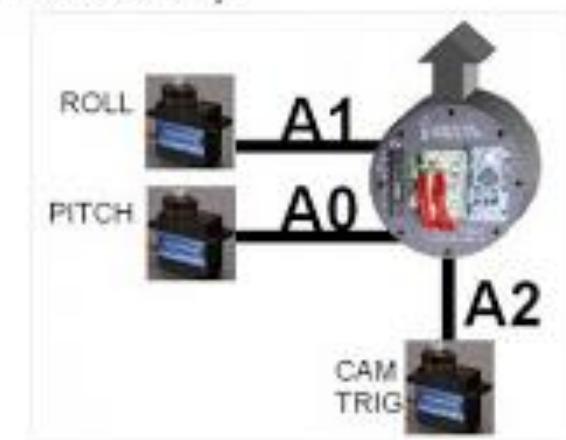
for arduino mega:



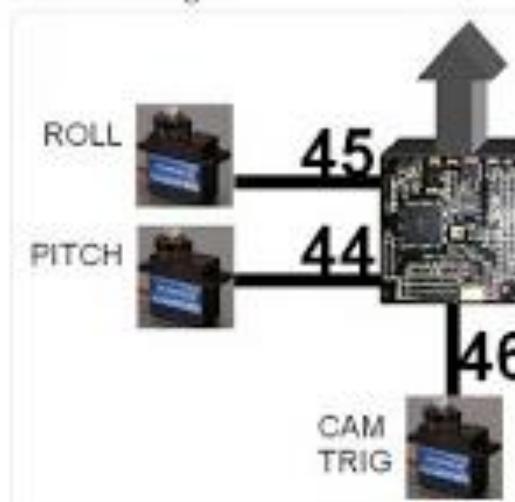
Pure stabilized gimbal system

For this setup, you need a GYRO + ACC setup (not possible with a gyro only setup)
A RC system is optional for this setup.

for Arduino 328p:



for arduino mega:



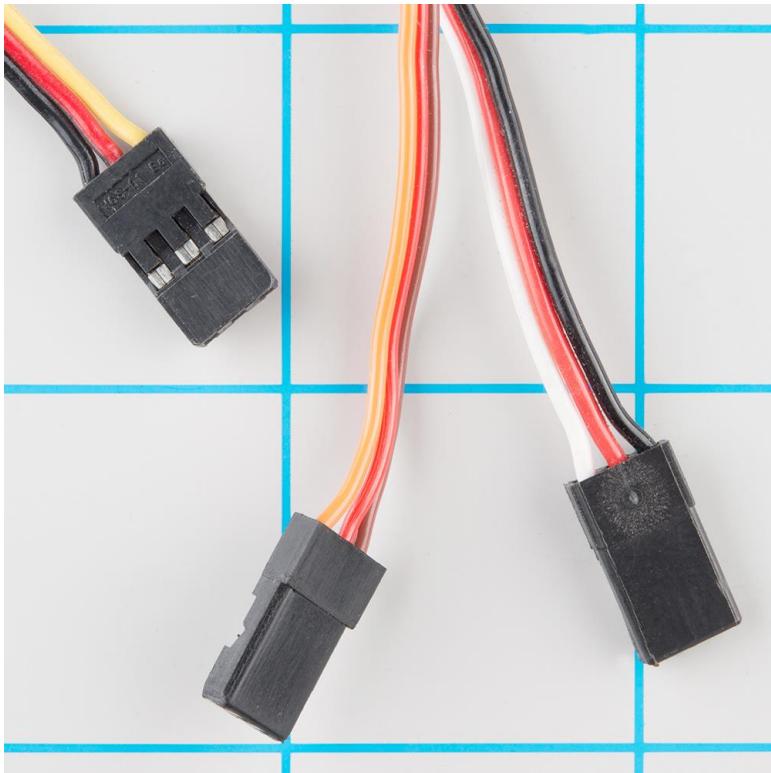
PWM INPUT Assignment

Pls Check the output pin from your Radio Rx manual



RX > Arduino / PWM in	Futaba Format	JR Format	Walkera Format	UNO 328 Input	Mega 2560 Input
Throttle	Ch3	Ch1	Ch3	D2	A8
Aileron	Ch1	Ch2	Ch2	D4	A9
Elevator	Ch2	Ch3	Ch1	D5	A10
Rudder	Ch4	Ch4	Ch4	D6	A11
Aux1	Ch5	Ch5	Ch5	D7	A12
Aux2	Ch6	Ch6	Ch6	D8	A13
Aux3	Ch7	Ch7	Ch7	N/A	A14
Aux4	Ch8	Ch8	Ch8	N/A	A15

SERVO HEADER



~ + -

End view

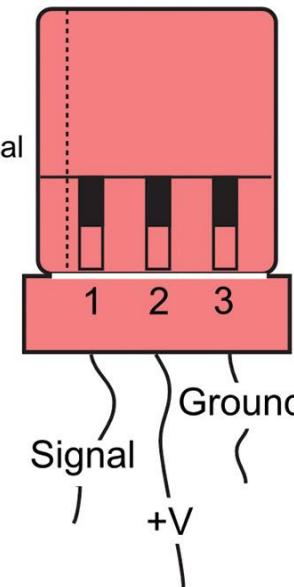


J-type (Futaba)



S-type (Hitec, JR)

Keyway =
signal terminal



They may come with different
coded wire but layout are
always same

OPTO Wires may only have
Signal and Negative Wires o



PWM RECEIVER

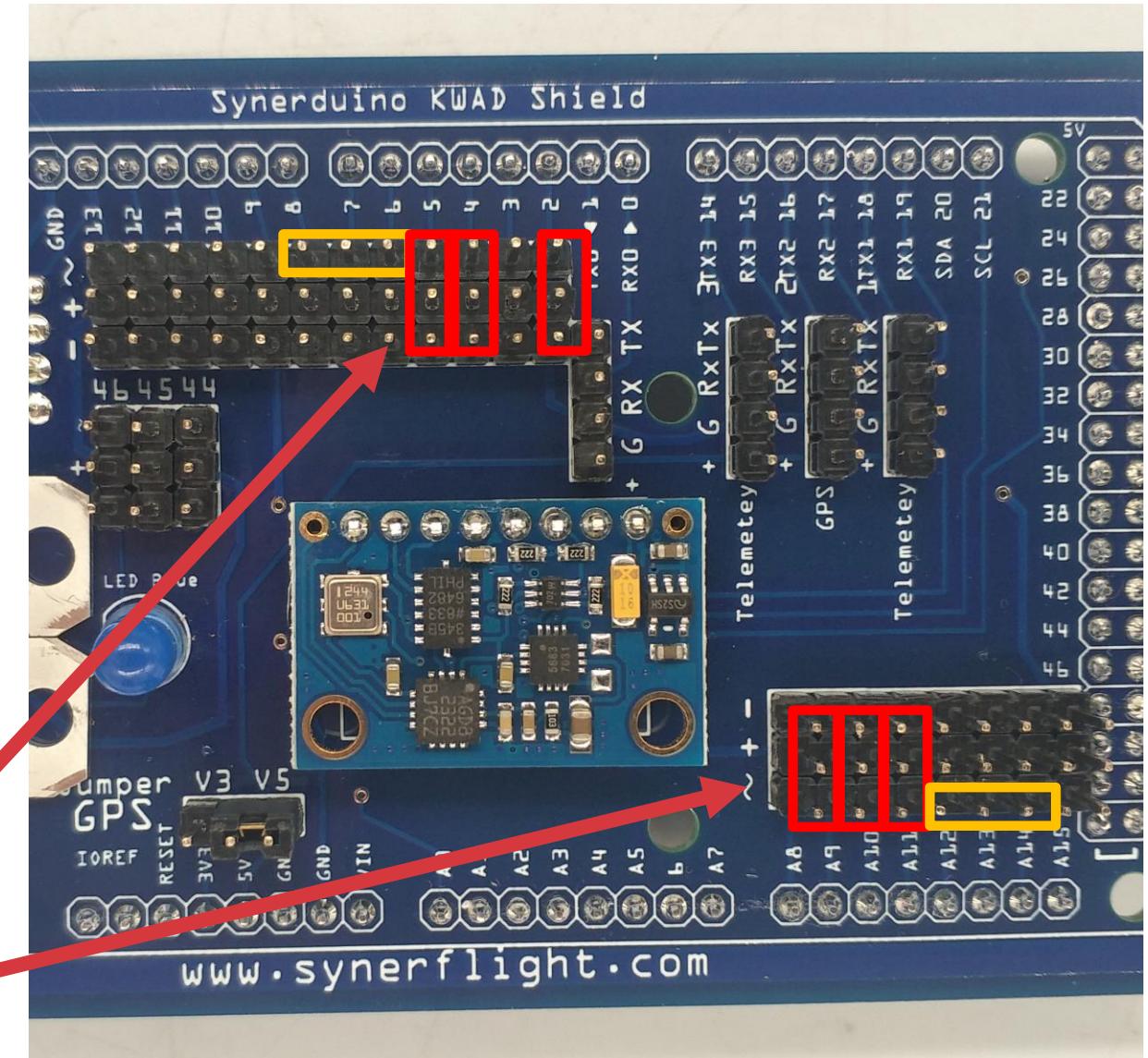


IN CASE YOU'D ASK WHY THE SERVO CONNECTORS WERE DONE THIS WAY , ITS SIMPLY YOU DON'T NEED TO PLUG ALL THE PWM POWER RAILS ON ALL CHANNELS YOU JUST NEED THE PWM SIGNAL PIN ALONE

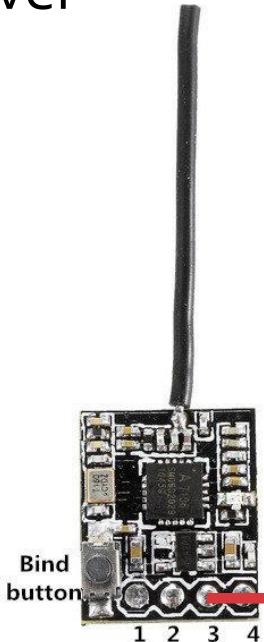
MOSTLY RUDDER AUX1 AND AUX2

UNO PWM IN

MEGA PWM IN



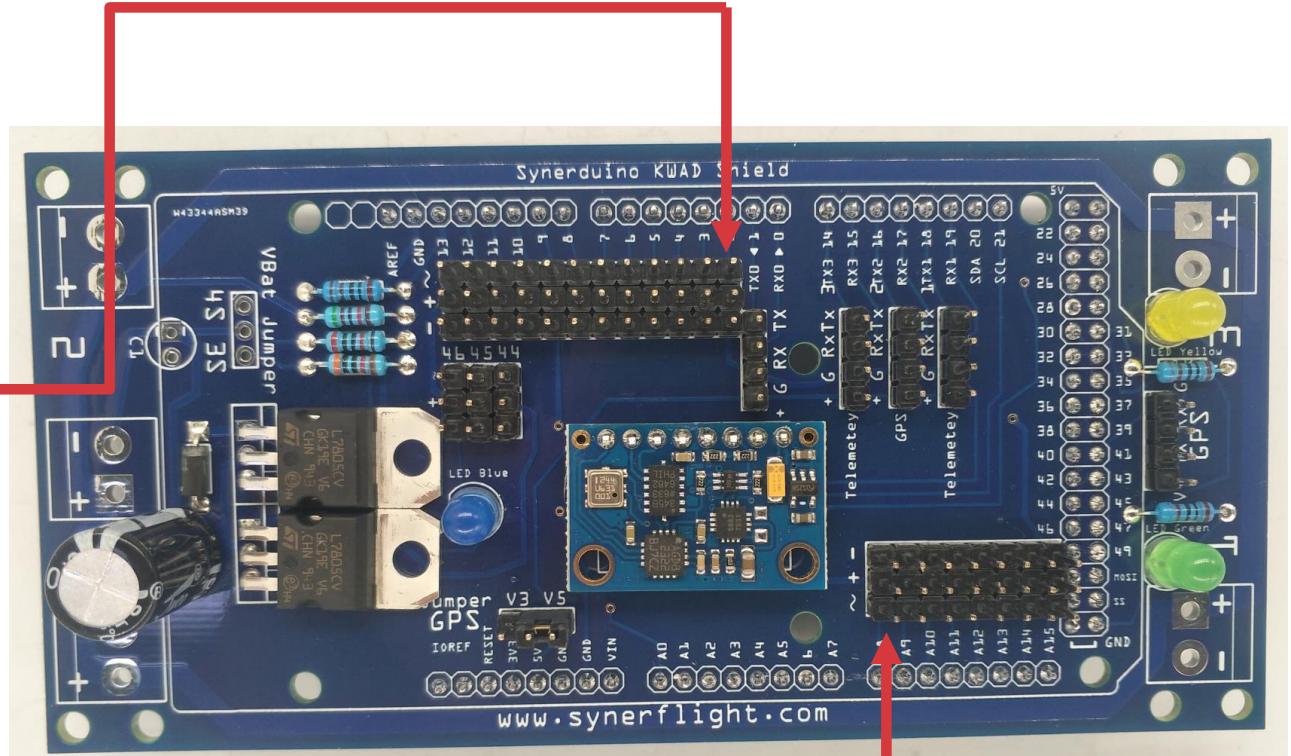
PPM Receiver



Pin D2 for Uno



Pin A8 for Mega



TELEMETRY



38400 OR 57600 FOR SIK
RADIO DEPENDING IF USES
433MHZ OR 900MHZ



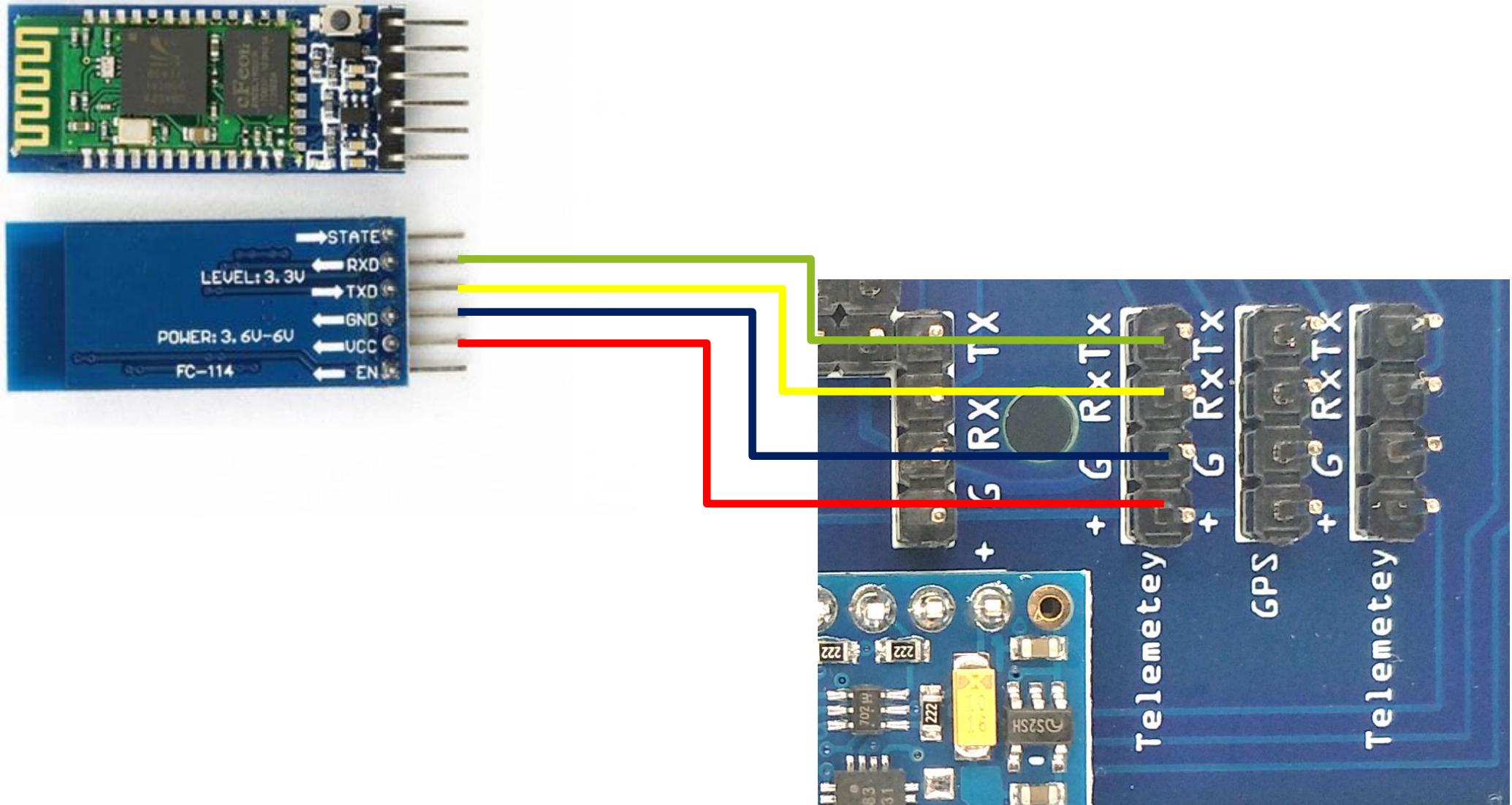
38400 FOR XBEE RADIO



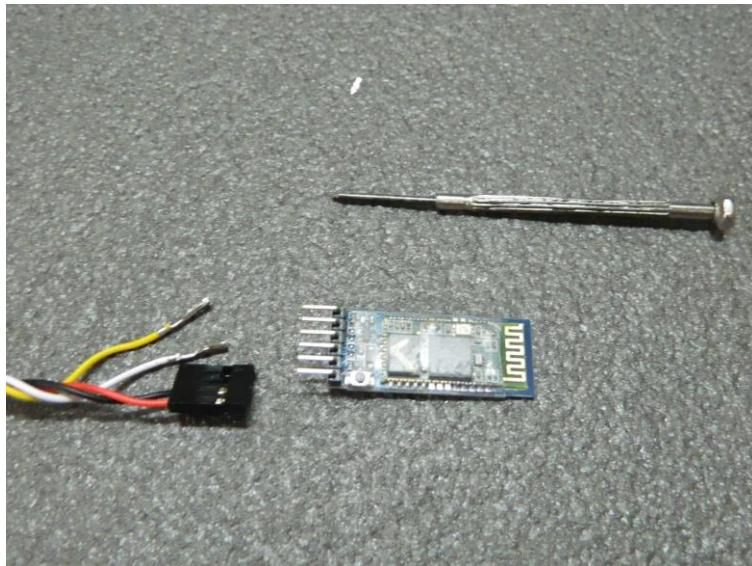
115200 FOR BLUETOOTH HC-05

STANDARD FOR ALL DRONES TO USE SERIAL LINK AS TELEMETRY MAINLY ON YOUR TX RX SERIAL PORTS , NOTE: THE LOWER THE FREQUENCY OF THE RADIO THE LOWER THE BAUD IS NEEDED
PROTOCOL IS MSP RAW OR MAVLINK

Bluetooth

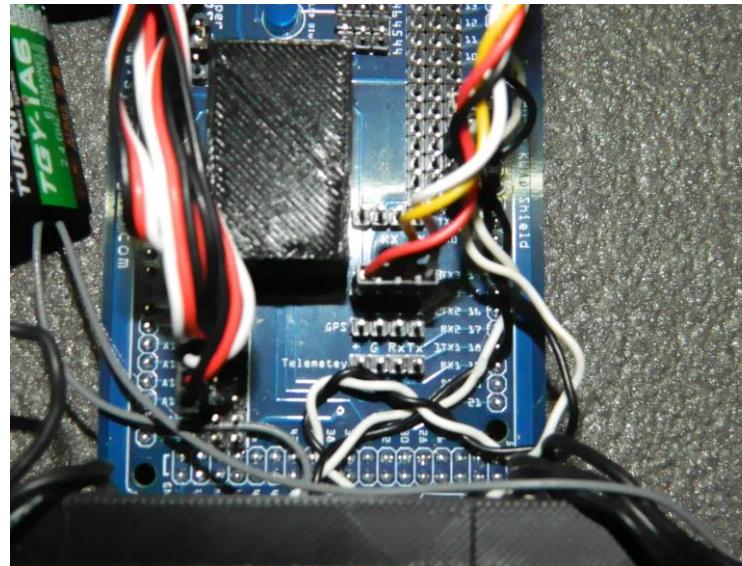


Bluetooth



**BLUETOOTH PLUG INTO SERIAL 1 OR
SERIAL 3**

115200 FOR BLUETOOTH HC-05



ATTENTION:

YOU MAY NEED TO REARRANGE THE HEADERS
TO CONNECT THE BLUETOOTH MODULE TO THE
SHIELD BOARD ACCORDINGLY

VCC >> +

GND >> G

TX >> RX

RX >> TX

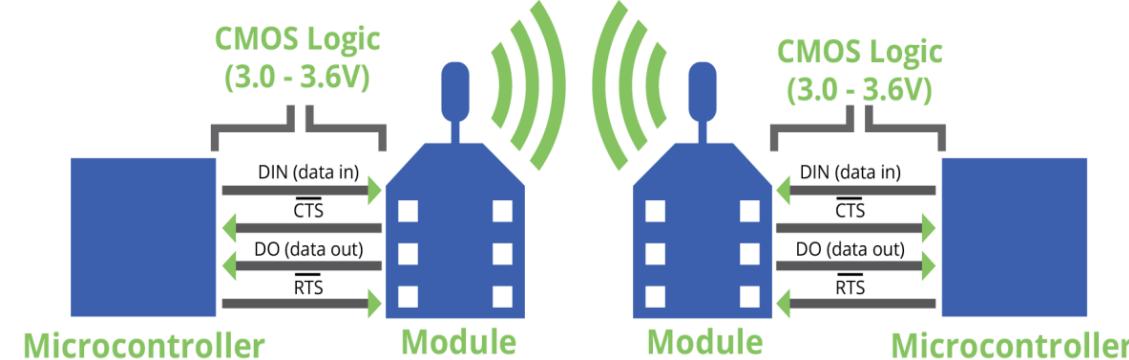
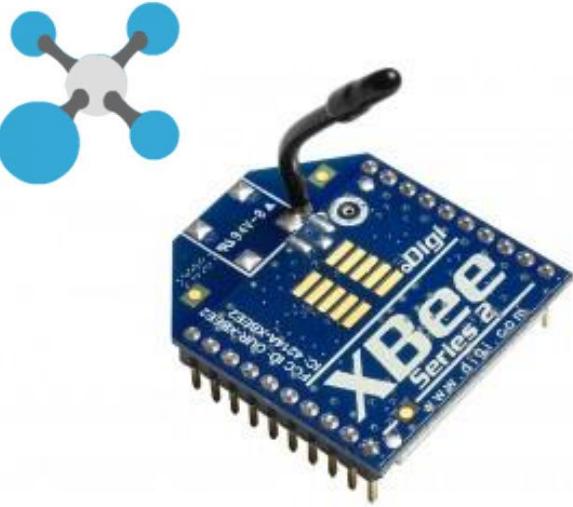


SEE TO IT THE WIRES COLOR CODE MATCHES THE
MARKINGS

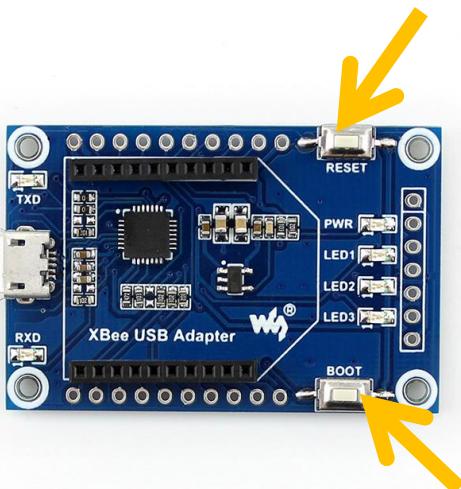
IMPROPER INSTALLATION MAY CAUSE DAMAGE
TO THE ARDUINO BOARD AND SHIELD DUE TO
REVERSE POLARITY

NOTE: WE PRESET THE BLUETOOTH FOR YOUR
CONVENIENCE TO THE PROPER SETUP BUT
SHOULD YOU WISH TO CHANGE THE SETTING ON
YOUR DIGRESSION

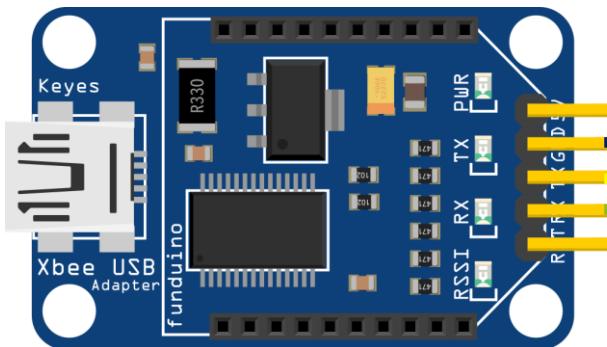
XBEE RADIO



38400 FOR XBEE RADIO



GET THE USB MODULE WITH
BOOT AND RESET BUTTON
AS YOU MAY NEED TO RESET
THE XBEE WHEN UPDATING
FIRMWARE

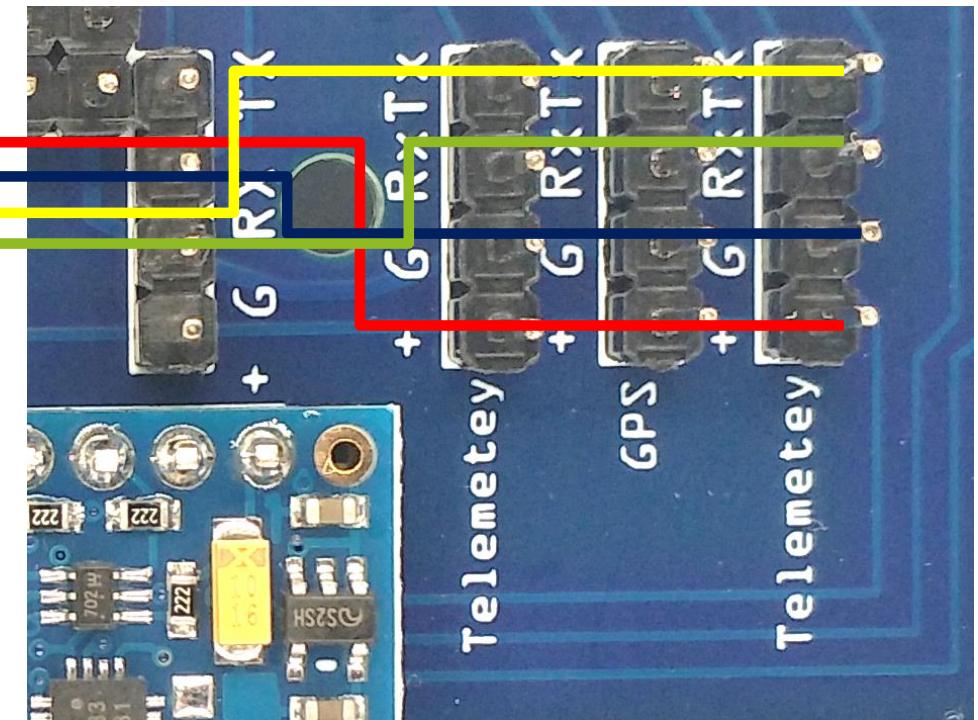


VCC >> +

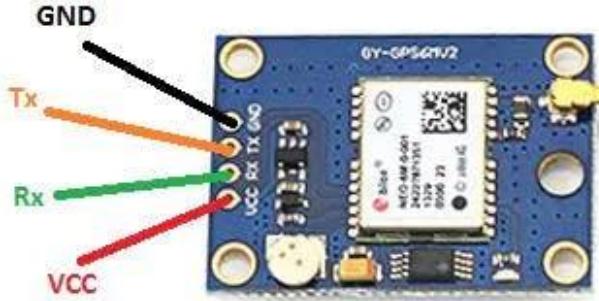
GND >> G

TX >> TX

RX >> RX



GPS



U BLOX NEO 6

PLUG IN TO SERIAL TX 2 RX 2 ON THE DRONE SHIELD BOARD

ATTENTION:

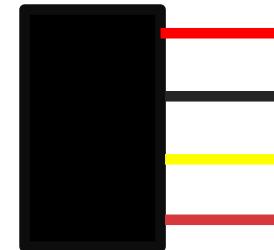
YOU MAY NEED TO REARRANGE THE HEADERS TO CONNECT THE GPS MODULE TO THE SHIELD BOARD ACCORDINGLY

VCC >> +

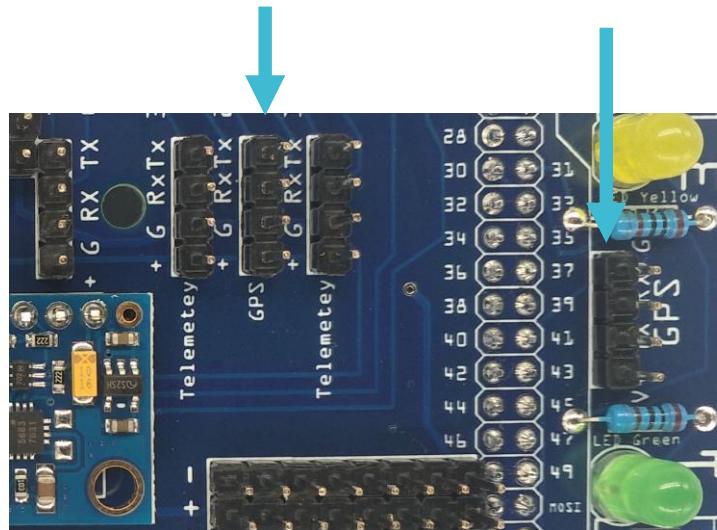
GND >> G

TX >> RX

RX >> TX



SEE TO IT THE WIRES COLOR CODE MATCHES THE MARKINGS

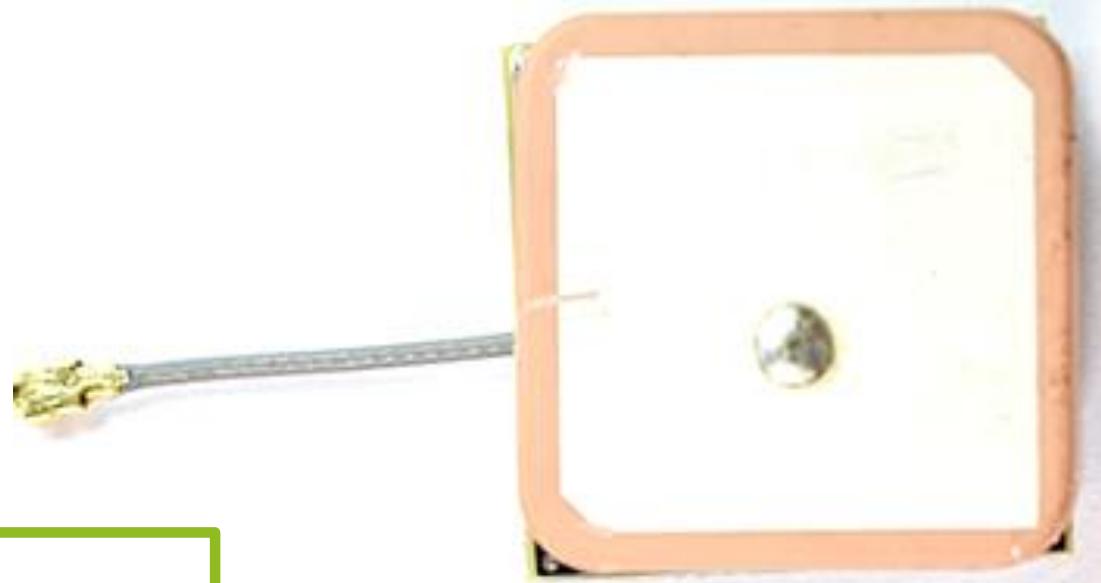
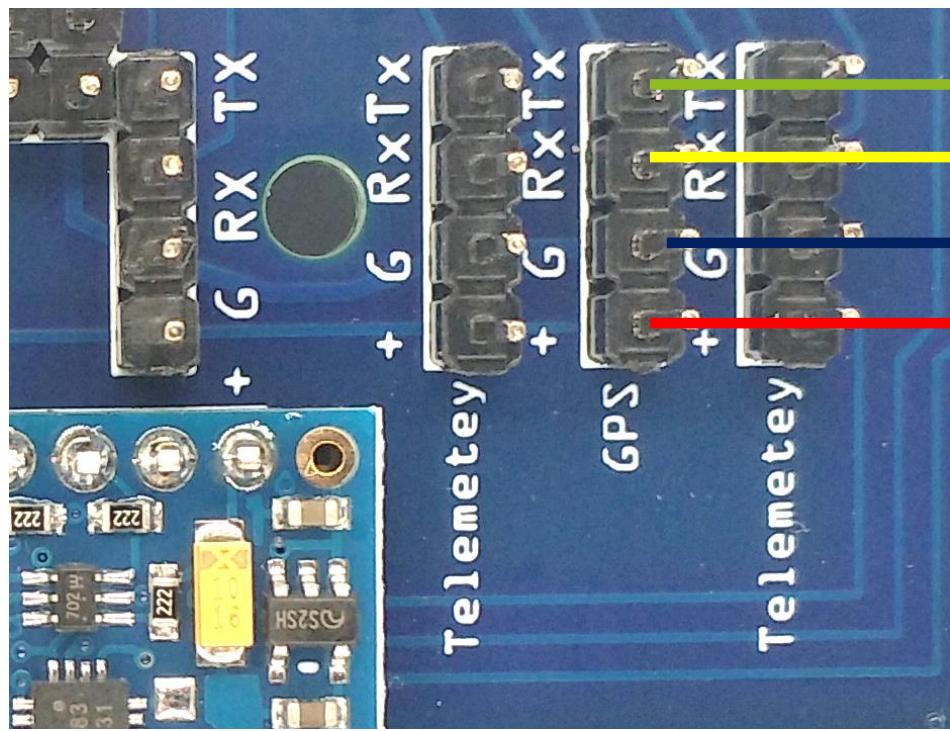


IMPROPER INSTALLATION MAY CAUSE DAMAGE TO THE ARDUINO BOARD AND SHIELD DUE TO REVERSE POLARITY

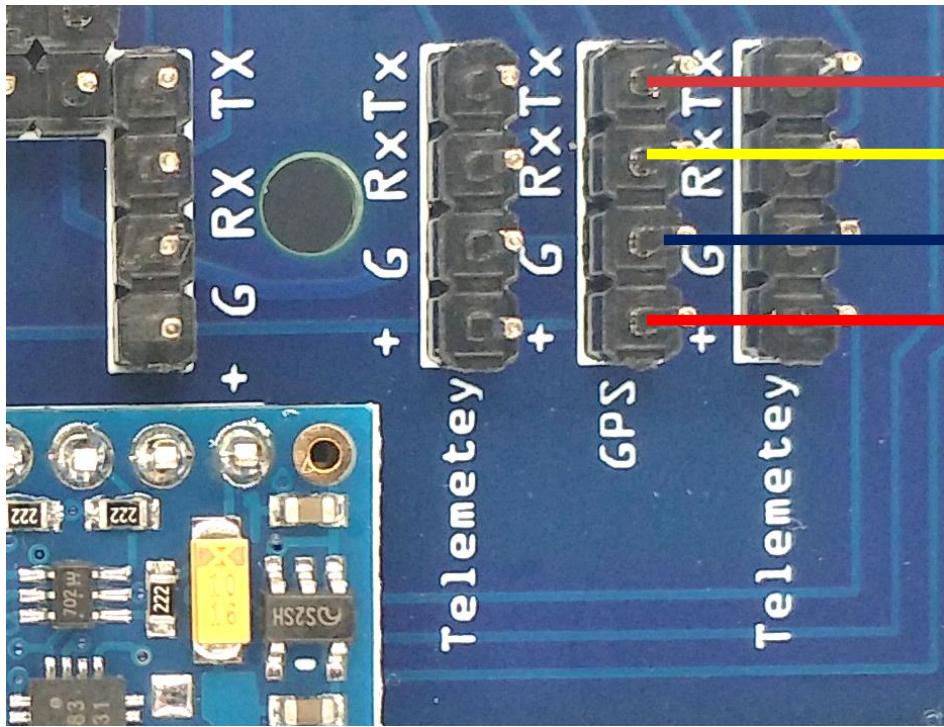
NOTE: YOU MAY NEED TO RE-SECURE THE GPS ANTENNA PATCH AGAIN WITH DOUBLE SIDED TAPE WHEN NECESSARY AS THE MODULE CAME IN WITH A TEMPORARY TAPE

NOTE: WE PRESET THE GPS FOR YOUR CONVENIENCE TO THE PROPER SETUP BUT SHOULD YOU WISH TO CHANGE THE SETTING ON YOUR DIGRESSION

GPS



GPS

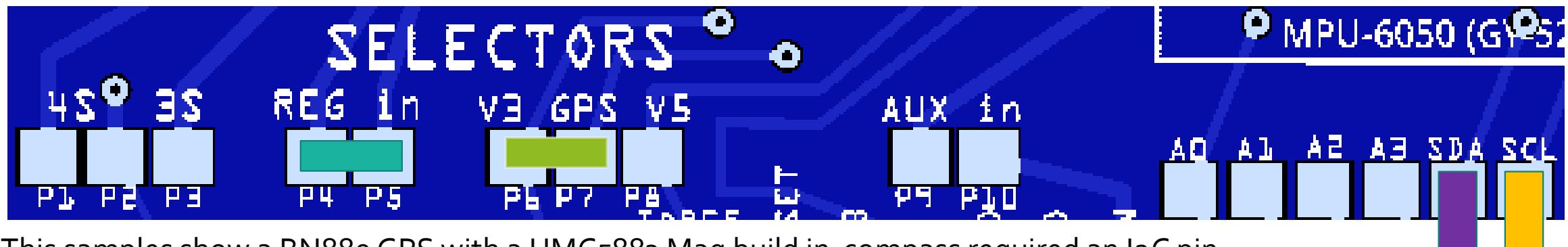


4.VCC
3.RX
2.TX
1.GND

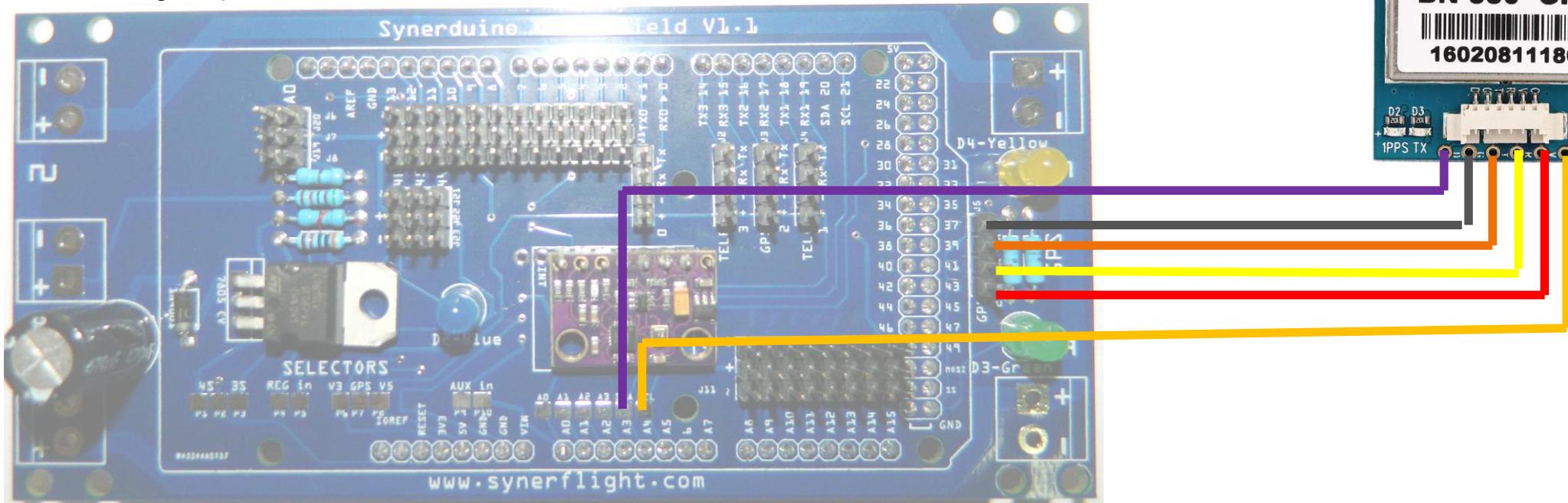
PIN	PIN Name	I/O	Description
1	GND	G	Ground
2	TX	O	Serial Data Output.
3	RX	I	Serial Data Input.
4	VCC	I	DC 3.0V - 5.5V supply input,Typical: 5.0V



External Sensors



This samples show a BN880 GPS with a HMC5883 Mag build in compass required an I₂C pin connection this works of all other I₂C sensors (pls ensure the address doesn't conflict with the IMU as found in Sensors.cpp) Note: other than the GPS build in sensors might require 3V you may need to set jumper to 3V





ARDUINO IDE

Application Needed

<https://www.arduino.cc/en/main/software>



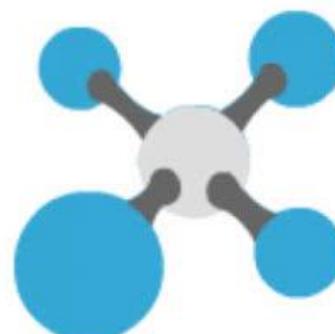
<http://synerflight.com/flywiogui/>

FLYWII GUI & Arduino Drone Multiwii firmware



UBLOX Configuration Platform

<https://www.u-blox.com/en/product/u-center>



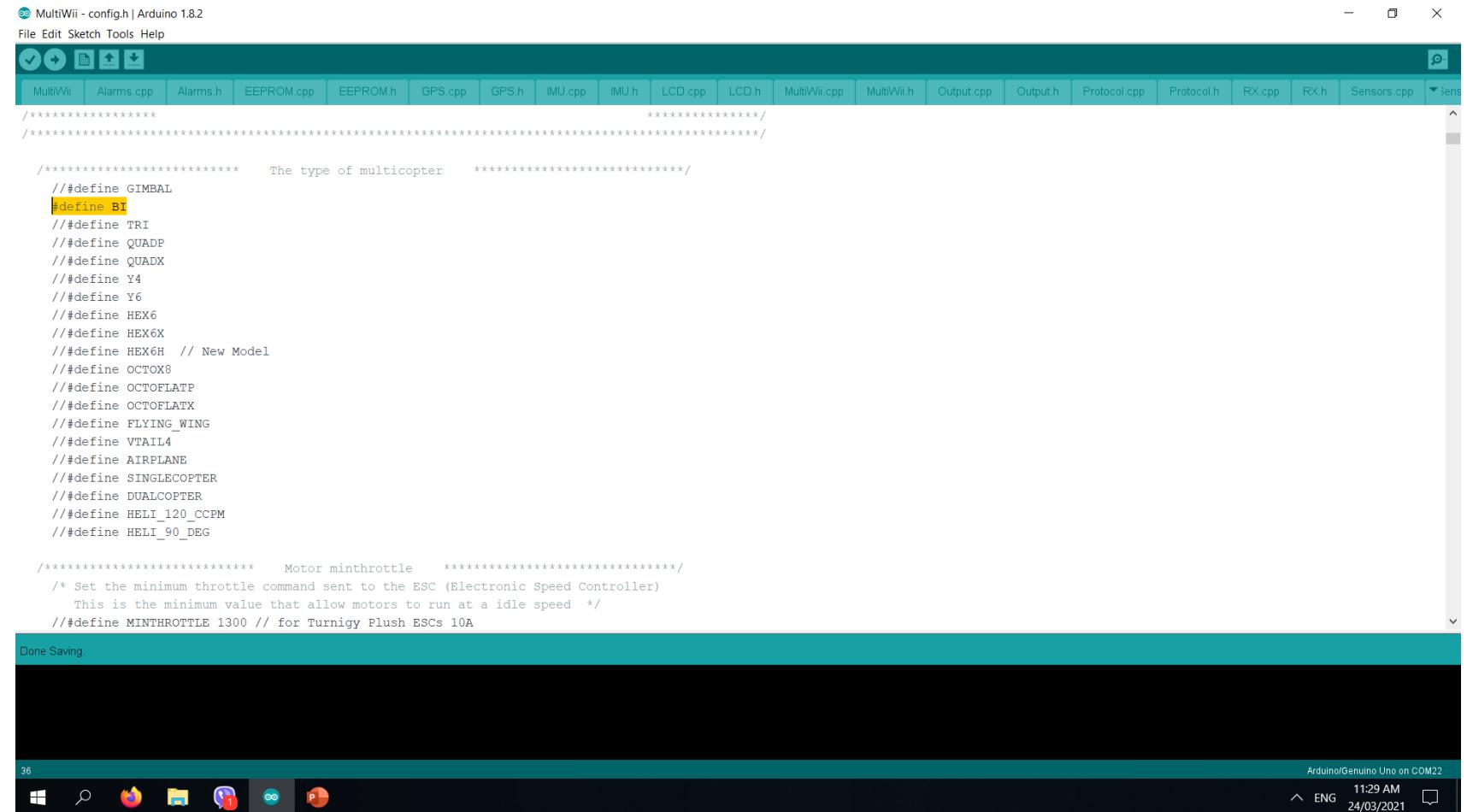
<https://www.digi.com/products/embedded-systems/digi-xbee/digi-xbee-tools/xctu>

XCTU Configuration Platform

Config.h Type of multirotor

Uncomment #Define Bi

This puts the vehicle into
Bicopter or surface vehicle mode



The screenshot shows the Arduino IDE interface with the file "config.h" open. The code editor displays configuration settings for different multirotor types. A specific line, "#define BI", is highlighted in yellow, indicating it is the target for uncommenting. The code also includes comments for various vehicle types like GIMBAL, TRI, QUADP, QUADX, Y4, Y6, HEX6, and HEX6H. It also defines OCTO8, OCTOFLATP, OCTOFLATX, FLYING_WING, VTAIL4, AIRPLANE, SINGLECOPTER, DUALCOPTER, HELI_120_CCPM, and HELI_90_DEG. A section at the bottom defines the minimum throttle command to the ESC, setting it to 1300 for Turnigy Plush ESCs at 10A.

```
MultiWii - config.h | Arduino 1.8.2
File Edit Sketch Tools Help
MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp Sens
/****************************************************************************
 * ***** The type of multicopter *****/
#ifndef GIMBAL
#define BI
#ifndef TRI
#ifndef QUADP
#ifndef QUADX
#ifndef Y4
#ifndef Y6
#ifndef HEX6
#ifndef HEX6H // New Model
#ifndef OCTO8
#ifndef OCTOFLATP
#ifndef OCTOFLATX
#ifndef FLYING_WING
#ifndef VTAIL4
#ifndef AIRPLANE
#ifndef SINGLECOPTER
#ifndef DUALCOPTER
#ifndef HELI_120_CCPM
#ifndef HELI_90_DEG
/*
 * ***** Motor minthrottle *****
 * Set the minimum throttle command sent to the ESC (Electronic Speed Controller)
 * This is the minimum value that allow motors to run at a idle speed */
#define MINTHROTTLE 1300 // for Turnigy Plush ESCs 10A
Done Saving.
Arduino/Genuino Uno on COM2
36
11:29 AM
24/03/2021
ENG

```

Config.h

Servos

Uncomment Disable servos when unarmed

Stop the motors from accidentally running when the vehicle is disarm

```
MultiWii - config.h | Arduino 1.8.2
File Edit Sketch Tools Help
Multivii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp sens

/*
 * NEW: not used anymore for servo coptertypes <== NEEDS FIXING - MOVE TO WIKI *
#define YAW_DIRECTION 1
//#define YAW_DIRECTION -1 // if you want to reverse the yaw correction direction

#define ONLYARMWHENFLAT //prevent the copter from arming when the copter is tilted

/********************* ARM/DISARM ********************/
/* optionally disable stick combinations to arm/disarm the motors.
 * In most cases one of the two options to arm/disarm via TX stick is sufficient */
#define ALLOW_ARM_DISARM_VIA_TX_YAW
//#define ALLOW_ARM_DISARM_VIA_TX_ROLL

/********************* SERVOS ********************/
/* info on which servos connect where and how to setup can be found here
 * http://www.multiwii.com/wiki/index.php?title=Config.h#Servos_configuration
 */

/* Do not move servos if copter is unarmed
 * It is a quick hack to overcome feedback tail wigglight when copter has a flexible
 * landing gear
*/
#define DISABLE_SERVOS_WHEN_UNARMED

/* if you want to preset min/middle/max values for servos right after flashing, because of limited physical
 * room for servo travel, then you must enable and set all three following options */
#define SERVO_MIN {1020, 1020, 1020, 1020, 1020, 1020, 1020}
#define SERVO_MAX {2000, 2000, 2000, 2000, 2000, 2000, 2000}
#define SERVO_MID {1500, 1500, 1500, 1500, 1500, 1500, 1500} // (*)
```

Done uploading.

avrduude done. Thank you.

255

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM31

ENG 5:24 PM 31/03/2021

Output.cpp

Mixtable

Motor reversing is done thru here apart those on the flywii GUI

Under ServoDir 4 & 5
(servo number , servo direction)

1=forward 2=reverse

```
MultiWii - Output.cpp | Arduino 1.8.2
File Edit Sketch Tools Help
MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.h MultiWii.cpp Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp Sensors.h

#define SERVODIR(n,b) ((conf.servoConf[n].rate & b) ? -1 : 1)

//***** main Mix Table *****
#if defined( MY_PRIVATE_MIXING )
#include MY_PRIVATE_MIXING
#elif defined( BI )
motor[0] = PIDMIX(+1, 0, 0); //LEFT
motor[1] = PIDMIX(-1, 0, 0); //RIGHT
servo[4] = +(SERVODIR(4,1) * axisPID[YAW]) + (SERVODIR(4,1) * axisPID[PITCH]) + get_middle(4); //LEFT
servo[5] = -(SERVODIR(5,2) * axisPID[YAW]) + (SERVODIR(5,1) * axisPID[PITCH]) + get_middle(5); //RIGHT
#elif defined( TRI )
motor[0] = PIDMIX( 0,+4/3, 0); //REAR
motor[1] = PIDMIX(-1,-2/3, 0); //RIGHT
motor[2] = PIDMIX(+1,-2/3, 0); //LEFT
servo[5] = (SERVODIR(5, 1) * axisPID[YAW]) + get_middle(5); //REAR
#elif defined( QUADP )
motor[0] = PIDMIX( 0,+1,-1); //REAR
motor[1] = PIDMIX(-1, 0,+1); //RIGHT
motor[2] = PIDMIX(+1, 0,+1); //LEFT
motor[3] = PIDMIX( 0,-1,-1); //FRONT
#elif defined( QUADX )
motor[0] = PIDMIX(-1,+1,-1); //REAR_R
motor[1] = PIDMIX(-1,-1,+1); //FRONT_R
motor[2] = PIDMIX(+1,+1,+1); //REAR_L
motor[3] = PIDMIX(+1,-1,-1); //FRONT_L
#elif defined( Y4 )
motor[0] = PIDMIX(+0,+1,-1); //REAR_1_CW
motor[1] = PIDMIX(-1,-1, 0); //FRONT_R_CCW
motor[2] = PIDMIX(+0,+1,+1); //REAR_2_CCW
#endif

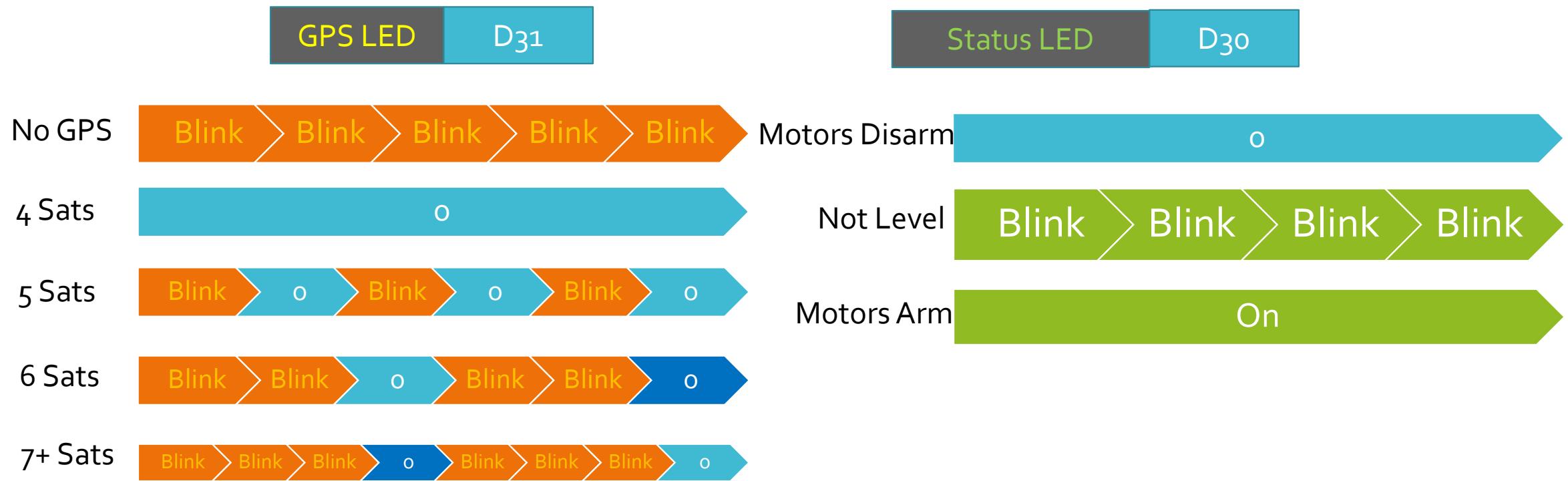
main Saving.

for Arduino 328p:

for arduino mega:

Arduino/Genuino Uno on COM22
11:26 AM
24/03/2021
```

LED Indicator



indicate a valid GPS fix by flashing the LED

- led work as sat number indicator
- No GPS FIX -> LED blinks constant speed
- Fix and sat no. below 5 -> LED off
- Fix and sat no. >= 5 -> LED blinks, one blink for 5 sat, two blinks for 6 sat, three for 7 +

INERTIAL MEASURING UNIT MEASURING UNIT

Please see the Board Specs
Data sheets for the installed
IMUs onboard

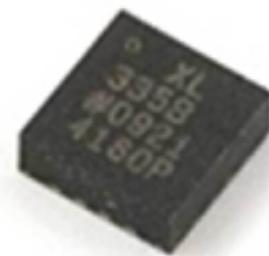


Magnetometer

Barometer

This is the heart of every flight controller AKA the
Main 4,

Gyro – stabilization on Roll Pitch Yaw Axis
Acc - Horizontal and Vertical stabilization XYZ
Baro – Altitude hold control
Mag – Heading and Compass



Accelerometer

Gyroscope

Each sensor has a corresponding address registry set
by manufacturer

You can find it on sensors.ccp tab

CONFIG.H



SynerduinoKwad3 - config.h | Arduino 1.8.5
File Edit Sketch Tools Help

SynerduinoKwad3 Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sens .cpp

```
***** constant loop time *****  
#define LOOP_TIME 2800  
  
***** boards and sensor definitions *****  
  
***** Combined IMU Boards *****  
/* if you use a specific sensor board:  
   please submit any correction to this list.  
   Note from Alex: I only own some boards, for other boards, I'm not sure, the info was gathered via rc forums, be cautious */  
##define FFIMUv1 // first 9DOF+baro board from Jussi, with HMC5843 <- confirmed by Alex  
##define FFIMUv2 // second version of 9DOF+baro board from Jussi, with HMC5883 <- confirmed by Alex  
##define FREEIMUv1 // v0.1 & v0.2 & v0.3 version of 9DOF board from Fabio  
##define FREEIMUv03 // FreeIMU v0.3 and v0.3.1  
##define FREEIMUv035 // FreeIMU v0.3.5 no baro  
##define FREEIMUv035_MS // FreeIMU v0.3.5_MS <- confirmed by Alex  
##define FREEIMUv035_BMP // FreeIMU v0.3.5_BMP  
##define FREEIMUv04 // FreeIMU v0.4 with MPU6050, HMC5883L, MS561101BA <- confirmed by Alex  
##define FREEIMUv043 // same as FREEIMUv04 with final MPU6050 (with the right ACC scale)  
##define NANOWII // the smallest multiwii FC based on MPU6050 + pro micro based proc <- confirmed by Alex  
##define PIPO // 9DOF board from erazz  
##define QUADRINO // full FC board 9DOF+baro board from witespy with BMP085 baro <- confirmed by Alex  
##define QUADRINO_ZOOM // full FC board 9DOF+baro board from witespy second edition  
##define QUADRINO_ZOOM_MS // full FC board 9DOF+baro board from witespy second edition <- confirmed by Alex  
##define ALLINONE // full FC board or standalone 9DOF+baro board from CSG_EU  
##define AEROQUADSHIELDv2  
##define ATAVRSBIN1 // Atmel 9DOF (Contribution by EOSBandi). requires 3.3V power. <- confirmed by Alex  
##define SIRIUS // Sirius Navigator IMU  
##define SIRIUSGPS // Sirius Navigator IMU using external MAG on GPS board <- confirmed by Alex
```

89 - 92

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM25
2:26 PM
^ ENG
30/09/2021

For the convenience of Synerduino Boards we have included a present on the sketch

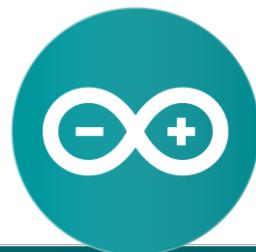
To use Combined IMU Boards
UnComment the # Synerduino board base on your version of the board and commend the // Independent Sensors//

//#define SYNERDUINO_GY801_V1

//#define SYNERDUINO_GY801_V2

#define SYNERDUINO_GY91_V1

DEF.H



SynerduinoKwad3 - def.h | Arduino 1.8.5

File Edit Sketch Tools Help

```
if defined(SYNERDUINO_GY801_V1)
#define L3G4200D // gyro
#define ADXL345 // acc
#define BMP085 // baro
#define MMC5883 // mag
#define ACC_ORIENTATION(X, Y, Z) {imu.accADC[ROLL] = -X; imu.accADC[PITCH] = -Y; imu.accADC[YAW] = Z;}
#define GYRO_ORIENTATION(X, Y, Z) {imu.gyroADC[ROLL] = Y; imu.gyroADC[PITCH] = -X; imu.gyroADC[YAW] = -Z;}
#define MAG_ORIENTATION(X, Y, Z) {imu.magADC[ROLL] = -X; imu.magADC[PITCH] = -Y; imu.magADC[YAW] = -Z;}
#undef INTERNAL_I2C_PULLUPS
#endif

#if defined(SYNERDUINO_GY801_V2)
#define L3G4200D // gyro
#define ADXL345 // acc
#define BMP085 // baro
#define MMC5883 // mag
#define ACC_ORIENTATION(X, Y, Z) {imu.accADC[ROLL] = -X; imu.accADC[PITCH] = -Y; imu.accADC[YAW] = Z;}
#define GYRO_ORIENTATION(X, Y, Z) {imu.gyroADC[ROLL] = Y; imu.gyroADC[PITCH] = -X; imu.gyroADC[YAW] = -Z;}
#define MAG_ORIENTATION(X, Y, Z) {imu.magADC[ROLL] = X; imu.magADC[PITCH] = Y; imu.magADC[YAW] = Z;}
#undef INTERNAL_I2C_PULLUPS
#endif

#if defined(SYNERDUINO_GY91_V1)
#define MPU6050 // gyro
#define BMP280 // baro
#define AK8963 // mag
#define ACC_ORIENTATION(X, Y, Z) {imu.accADC[ROLL] = -X; imu.accADC[PITCH] = -Y; imu.accADC[YAW] = Z;}
#define GYRO_ORIENTATION(X, Y, Z) {imu.gyroADC[ROLL] = Y; imu.gyroADC[PITCH] = -X; imu.gyroADC[YAW] = -Z;}
#define MAG_ORIENTATION(X, Y, Z) {imu.magADC[ROLL] = Y; imu.magADC[PITCH] = Y; imu.magADC[YAW] = -Z;}

```

Board Define orientation can be found in Def.h Tab

IMU Orientations and Sensor definitions

1671 - 1641

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM25

2:33 PM
ENG
30/09/2021

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help



```
/* leave it commented if you already checked a specific board above */  
/* I2C gyroscope */
```

```
//#define WMP  
//#define ITG3050  
//#define ITG3200  
//#define MPU3050  
#define L3G4200D  
##define MPU6050      //combo + ACC  
##define LSM330        //combo + ACC
```

```
/* I2C accelerometer */  
##define MMA7455  
#define ADXL345  
##define BMA020  
##define BMA180  
##define BMA280  
##define LIS3LV02  
##define LSM303DLX_ACC  
##define MMA8451Q
```

```
/* I2C barometer */  
#define BMP085  
##define MS561101BA
```

```
/* I2C magnetometer */  
##define HMC5843  
#define HMC5883  
##define HMC5883L
```

Depending on the Version of ic2 sensors installed on the board
you got select appropriately for it to work

To use independent sensor **Comment the // Combined IMU Boards // and uncomment the # Independent Sensors**

Synerduino GY801

```
#define L3G4200D // gyro  
#define ADXL345 // acc  
#define BMP085 // baro  
#define MMC5883 // mag
```

Synerduino GY91

```
#define MPU6050 // gyro  
#define BMP280 // baro  
#define AK8963 // mag
```

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp ▾ Sens

```
***** independent sensors *****

/* leave it commented if you already checked a specific board above */

/* I2C gyroscope */
#ifndef WMP
#define ITG3050
#ifndef ITG3200
#define MPU3050
#define L3G4200D
#define MPU6050 //combo + ACC
#define LSM330 //combo + ACC

/* I2C accelerometer */
#define MMA7455
#define ADXL345
#define BMA020
#define BMA180
#define BMA280
#define LIS3LV02
#define LSM303DLX_ACC
#define MMA8451Q

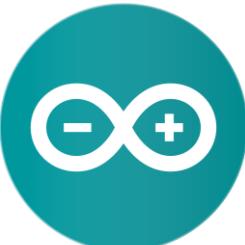
/* I2C barometer */
#define BMP085
#define MS561101BA

/* I2C magnetometer */
#define HMC5843
#define HMC5883
//<--> ->-->
```

Depending on the Version of ic2 sensors installed on the board
you got select appropriately for it to work

163 Arduino/Genuino Uno on COM41 9:28 PM 13/02/2020

SENSORS.CCP



MultiWii - Sensors.cpp | Arduino 1.8.2

File Edit Sketch Tools Help

MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp

```
#endif

// ****
// I2C Gyroscope L3G4200D
// ****

#if defined(L3G4200D)
#define L3G4200D_ADDRESS 0x69
void Gyro_init() {
    delay(100);
    i2c_writeReg(L3G4200D_ADDRESS, 0x20, 0x8F); // CTRL_REG1 400Hz ODR, 20hz filter, run!
    delay(5);
    i2c_writeReg(L3G4200D_ADDRESS, 0x24, 0x02); // CTRL_REG5 low pass filter enable
    delay(5);
    i2c_writeReg(L3G4200D_ADDRESS, 0x23, 0x30); // CTRL_REG4 Select 2000dps
}

void Gyro_getADC () {
    i2c_getSixRawADC(L3G4200D_ADDRESS, 0x80|0x28);

    GYRO_ORIENTATION( ((rawADC[1]<<8) | rawADC[0])>>2 ,
                      ((rawADC[3]<<8) | rawADC[2])>>2 ,
                      ((rawADC[5]<<8) | rawADC[4])>>2 );
    GYRO_Common();
}
#endif

// ****
// I2C Gyroscope ITG3200 / ITG3205 / ITG3050 / MPU3050
<
```

851 Arduino/Genuino Uno on COM41

9:37 PM ENG 13/02/2020

SENSORS.CCP



The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** MultiWii - Sensors.cpp | Arduino 1.8.2
- Menu Bar:** File Edit Sketch Tools Help
- Toolbar:** Includes icons for Save, Undo, Redo, Open, Upload, Download, and a magnifying glass.
- Sketch List:** A horizontal bar at the top with tabs for various files: MultiWii, Alarms.cpp, Alarms.h, EEPROM.cpp, EEPROM.h, GPS.cpp, GPS.h, IMU.cpp, IMU.h, LCD.cpp, LCD.h, MultiWii.cpp, MultiWii.h, Output.cpp, Output.h, Protocol.cpp, Protocol.h, RX.cpp, RX.h, Sensors.cpp. The Sensors.cpp tab is currently selected.
- Code Editor:** The main area displays the C++ code for the Sensors sketch. The code includes comments explaining the I2C setup for the ADXL345 accelerometer, pin assignments, and initialization functions for both the Accelerometer and Gyroscope.

```
// I2C Accelerometer ADXL345
// ****
// I2C adress: 0x3A (8bit)    0x1D (7bit)
// Resolution: 10bit (Full range - 14bit, but this is autoscaling 10bit ADC to the range +- 16g)
// principle:
// 1) CS PIN must be linked to VCC to select the I2C mode
// 2) SDO PIN must be linked to VCC to select the right I2C adress
// 3) bit b00000100 must be set on register 0x2D to read data (only once at the initialization)
// 4) bits b00001011 must be set on register 0x31 to select the data format (only once at the initialization)
// ****

#if defined(ADXL345)
#if !defined(ADXL345_ADDRESS)
#define ADXL345_ADDRESS 0x1D
#define ADXL345_ADDRESS 0x53 //WARNING: Conflicts with a Wii Motion plus!
#endif

void ACC_init () {
    delay(10);
    i2c_writeReg(ADXL345_ADDRESS,0x2D,1<<3); // register: Power CTRL -- value: Set measure bit 3 on
    i2c_writeReg(ADXL345_ADDRESS,0x31,0x0B); // register: DATA_FORMAT -- value: Set bits 3(full range) and 1 0 on (+/- 16g-range)
    i2c_writeReg(ADXL345_ADDRESS,0x2C,0x09); // register: BW_RATE -- value: rate=50hz, bw=20hz
}

void ACC_getADC () {
    i2c_getSixRawADC(ADXL345_ADDRESS,0x32);

    ACC_ORIENTATION( ((rawADC[1]<<8) | rawADC[0]) ,
                    ((rawADC[3]<<8) | rawADC[2]) ,
                    ((rawADC[5]<<8) | rawADC[4]) );
}
```

SENSORS.CCP

MultiWii - Sensors.cpp | Arduino 1.8.2

File Edit Sketch Tools Help

The image shows the Arduino IDE interface with the file "Sensors.cpp" open. The code is for a BOSCH BMP085 I2C Barometer sensor. It includes comments explaining the calibration process (steps 1-4) and defines a struct for sensor registers. The code uses union types to handle raw byte data and calculate compensated temperature and pressure. A note at the bottom indicates transforming big-endian bytes to little-endian.

```
// ****
// I2C Barometer BOSCH BMP085
// ****
// I2C adress: 0x77 (7bit)
// principle:
// 1) read the calibration register (only once at the initialization)
// 2) read uncompensated temperature (not mandatory at every cycle)
// 3) read uncompensated pressure
// 4) raw temp + raw pressure => calculation of the adjusted pressure
// the following code uses the maximum precision setting (oversampling setting 3)
// ****

#if defined(BMP085)
#define BMP085_ADDRESS 0x77

static struct {
    // sensor registers from the BOSCH BMP085 datasheet
    int16_t ac1, ac2, ac3;
    uint16_t ac4, ac5, ac6;
    int16_t b1, b2, mb, mc, md;
    union {uint16_t val; uint8_t raw[2]; } ut; //uncompensated T
    union {uint32_t val; uint8_t raw[4]; } up; //uncompensated P
    uint8_t state;
    uint32_t deadline;
} bmp085_ctx;
#define OSS 3

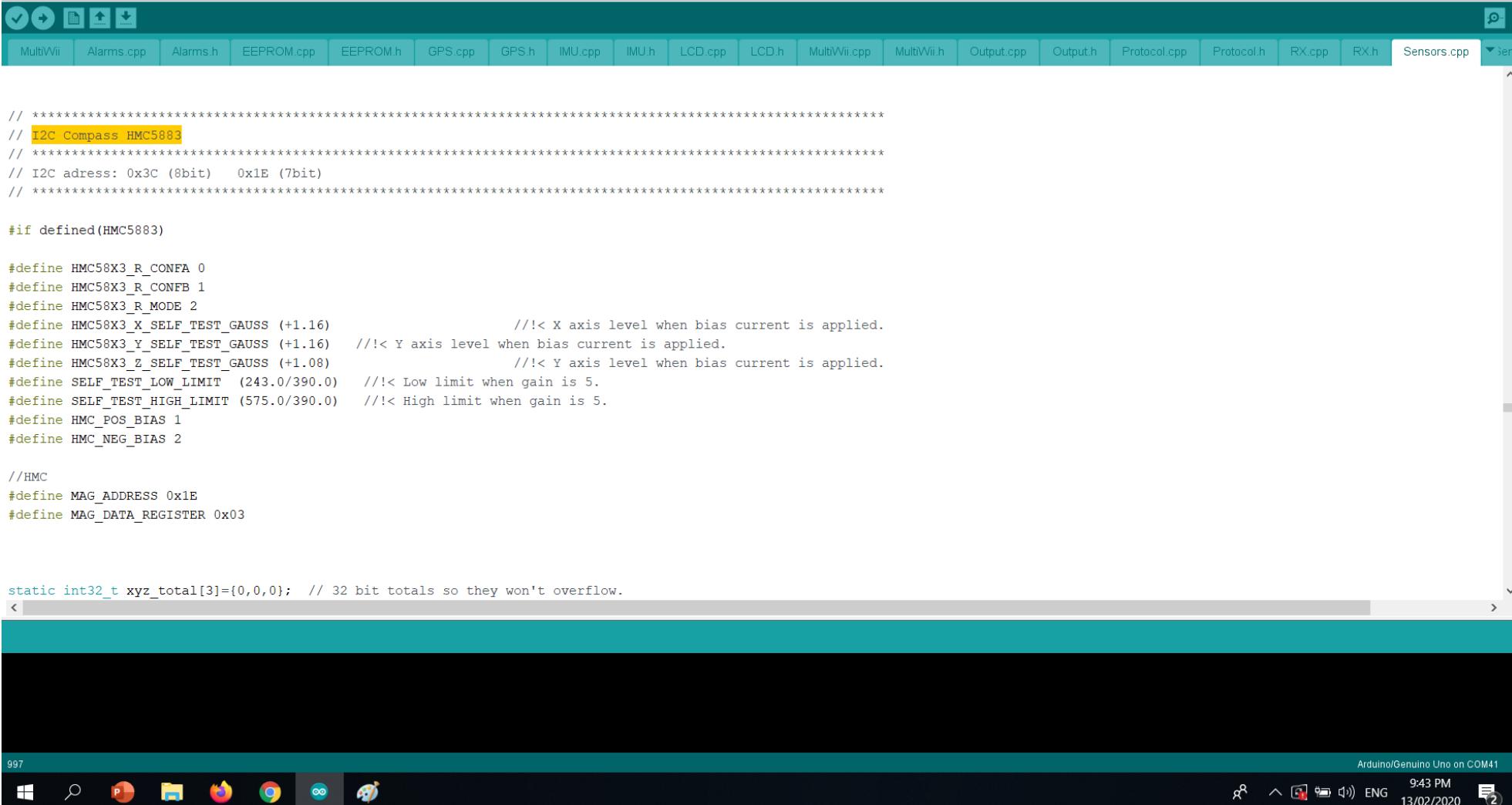
/* transform a series of bytes from big endian to little
<
```

331 Arduino/Genuino Uno on COM41

Windows taskbar icons: File Explorer, Edge, Google Chrome, Paint 3D, Task View, Start button.

System tray icons: Battery, Signal strength, Volume, ENG, 9:40 PM, 13/02/2020, Notifications (2).

SENSORS.CPP



The image shows the Arduino IDE interface with the file "Sensors.cpp" open. The title bar reads "MultiWii - Sensors.cpp | Arduino 1.8.2". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for save, build, upload, and refresh. The tab bar shows multiple files: MultiWii, Alarms.cpp, Alarms.h, EEPROM.cpp, EEPROM.h, GPS.cpp, GPS.h, IMU.cpp, IMU.h, LCD.cpp, LCD.h, MultiWii.cpp, MultiWii.h, Output.cpp, Output.h, Protocol.cpp, Protocol.h, RX.cpp, RX.h, Sensors.cpp, and sens. The code editor displays the following C++ code:

```
// **** I2C Compass HMC5883 ****
// **** I2C adress: 0x3C (8bit)  0x1E (7bit)
// ****

#if defined(HMC5883)

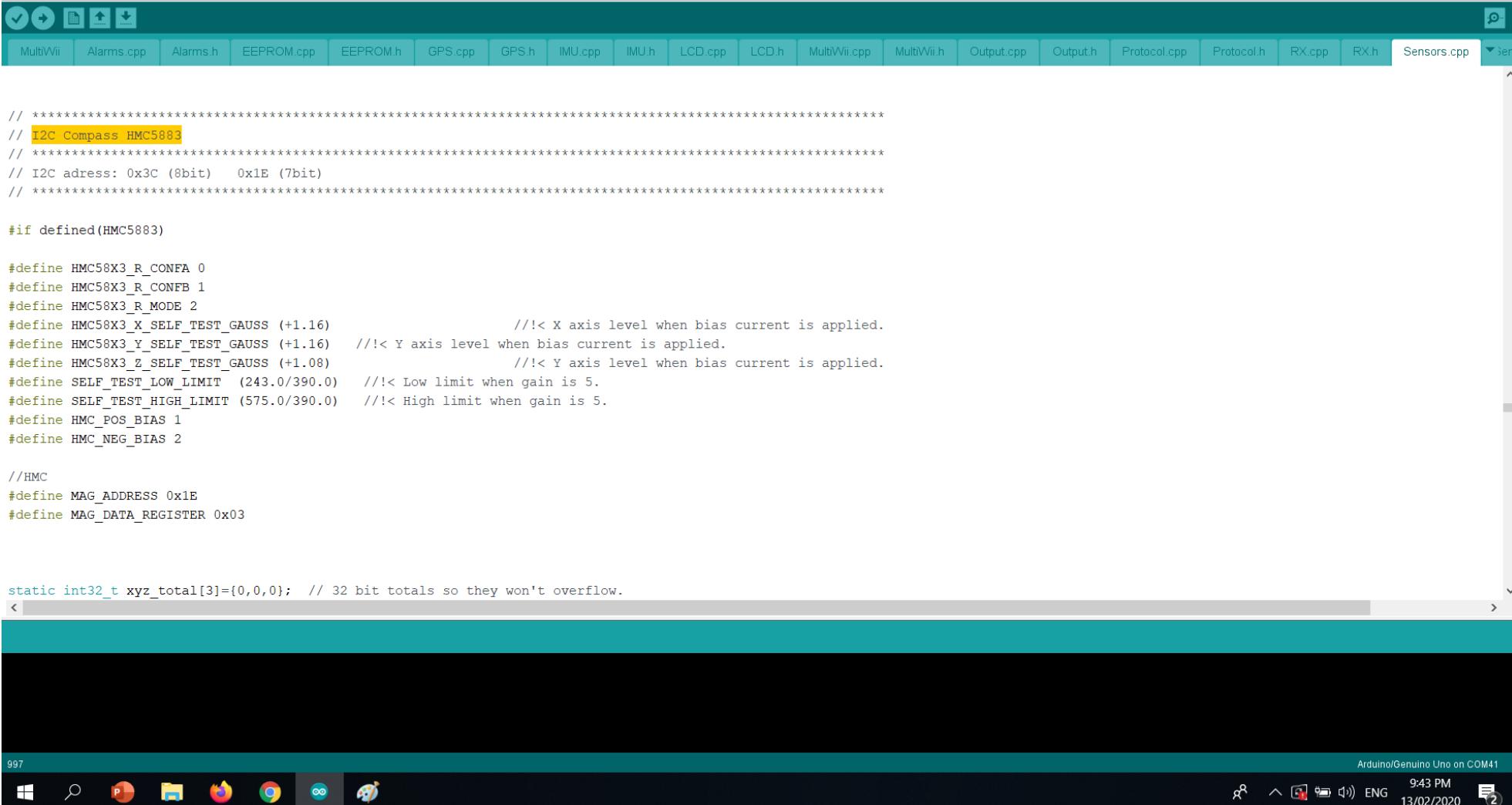
#define HMC58X3_R_CONFA 0
#define HMC58X3_R_CONFB 1
#define HMC58X3_R_MODE 2
#define HMC58X3_X_SELF_TEST_GAUSS (+1.16)           //!< X axis level when bias current is applied.
#define HMC58X3_Y_SELF_TEST_GAUSS (+1.16)           //!< Y axis level when bias current is applied.
#define HMC58X3_Z_SELF_TEST_GAUSS (+1.08)            //!< Z axis level when bias current is applied.
#define SELF_TEST_LOW_LIMIT (243.0/390.0)           //!< Low limit when gain is 5.
#define SELF_TEST_HIGH_LIMIT (575.0/390.0)           //!< High limit when gain is 5.
#define HMC_POS_BIAS 1
#define HMC_NEG_BIAS 2

//HMC
#define MAG_ADDRESS 0x1E
#define MAG_DATA_REGISTER 0x03

static int32_t xyz_total[3]={0,0,0}; // 32 bit totals so they won't overflow.
```

The status bar at the bottom indicates "997" (line count), "Arduino/Genuino Uno on COM41", "943 PM", "ENG", "13/02/2020", and a battery icon with a "2" inside.

SENSORS.CPP



The image shows the Arduino IDE interface with the file "Sensors.cpp" open. The title bar reads "MultiWii - Sensors.cpp | Arduino 1.8.2". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar has icons for save, build, and upload. The tab bar shows multiple files: MultiWii, Alarms.cpp, Alarms.h, EEPROM.cpp, EEPROM.h, GPS.cpp, GPS.h, IMU.cpp, IMU.h, LCD.cpp, LCD.h, MultiWii.cpp, MultiWii.h, Output.cpp, Output.h, Protocol.cpp, Protocol.h, RX.cpp, RX.h, Sensors.cpp, and sens. The code editor displays the following C++ code:

```
// **** I2C Compass HMC5883 ****
// **** I2C adress: 0x3C (8bit)  0x1E (7bit)
// ****

#if defined(HMC5883)

#define HMC58X3_R_CONFA 0
#define HMC58X3_R_CONFB 1
#define HMC58X3_R_MODE 2
#define HMC58X3_X_SELF_TEST_GAUSS (+1.16)           //!< X axis level when bias current is applied.
#define HMC58X3_Y_SELF_TEST_GAUSS (+1.16)           //!< Y axis level when bias current is applied.
#define HMC58X3_Z_SELF_TEST_GAUSS (+1.08)            //!< Z axis level when bias current is applied.
#define SELF_TEST_LOW_LIMIT (243.0/390.0)           //!< Low limit when gain is 5.
#define SELF_TEST_HIGH_LIMIT (575.0/390.0)           //!< High limit when gain is 5.
#define HMC_POS_BIAS 1
#define HMC_NEG_BIAS 2

//HMC
#define MAG_ADDRESS 0x1E
#define MAG_DATA_REGISTER 0x03

static int32_t xyz_total[3]={0,0,0}; // 32 bit totals so they won't overflow.
```

The status bar at the bottom indicates "997" (line count), "Arduino/Genuino Uno on COM41", "943 PM", "ENG", "13/02/2020", and a message center icon with a "2". The taskbar at the very bottom shows the Windows Start button, a search icon, a folder icon, a Firefox icon, a Chrome icon, the Arduino icon, and a globe icon.

SENSOR ORIENTATION

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

Multivii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp Sensors.h

```
//#define HMC5843
#define HMC5883
#define AK8975
#define MAG3110

/* Sonar */ // for visualization purpose currently - no control code behind
#define SRF02 // use the Devantech SRF i2c sensors
#define SRF08
#define SRF10
#define SRF23

/* ADC accelerometer */ // for 5DOF from sparkfun, uses analog PIN A1/A2/A3
#define ADCACC

/* enforce your individual sensor orientation - even overrides board specific defaults */
#define FORCE_ACC_ORIENTATION(X, Y, Z) {imu.accADC[ROLL] = X; imu.accADC[PITCH] = Y; imu.accADC[YAW] = Z;}
#define FORCE_GYRO_ORIENTATION(X, Y, Z) {imu.gyroADC[ROLL] = -Y; imu.gyroADC[PITCH] = X; imu.gyroADC[YAW] = -Z;}
#define FORCE_MAG_ORIENTATION(X, Y, Z) {imu.magADC[ROLL] = -X; imu.magADC[PITCH] = -Y; imu.magADC[YAW] = Z;}

/* Board orientation shift */
/* If you have frame designed only for + mode and you cannot rotate FC physically for flying in X mode (or vice versa)
 * you can use one of this options for virtual sensors rotation by 45 degrees, then set type of multicopter according to flight mode.
 * Check motors order and directions of motors rotation for matching with new front point! Uncomment only one option! */
#define SENSORS_TILT_45DEG_RIGHT // rotate the FRONT 45 degrees clockwise
#define SENSORS_TILT_45DEG_LEFT // rotate the FRONT 45 degrees counterclockwise

//*****
```

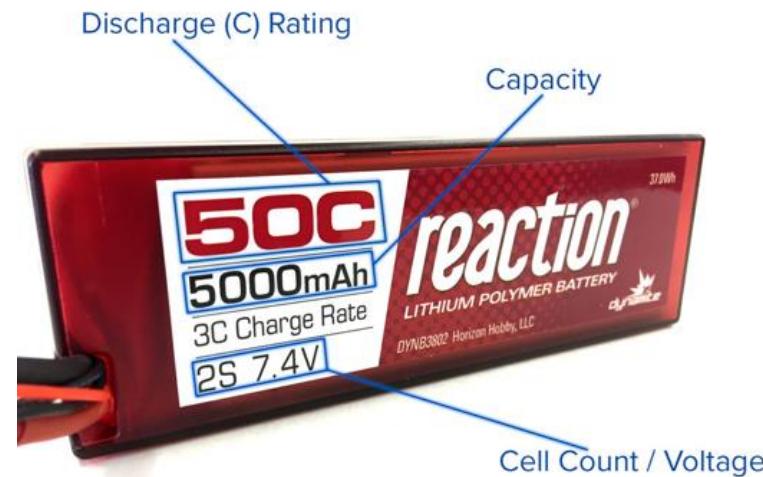
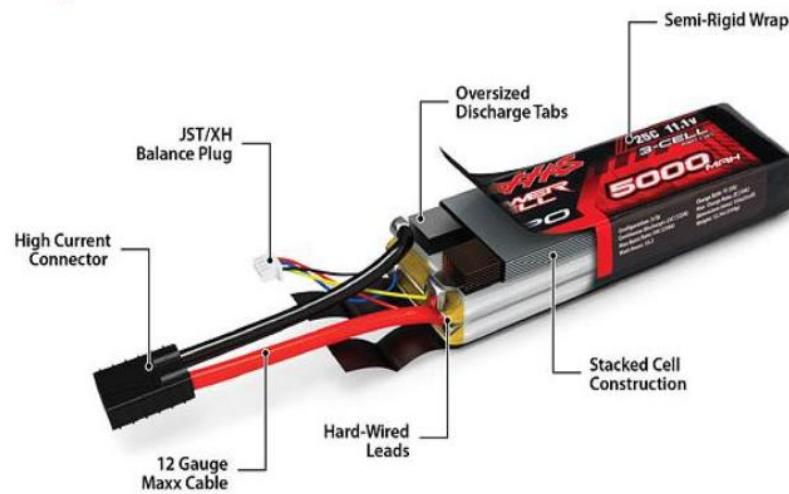
203 - 206

Arduino/Ger 2 new notifications

9:33 PM 13/02/2020

**UNCOMMENTING THESE LINES
FORCES THE ORIENTATION TO
ENFORCE INDIVIDUAL SENSOR
ORIENTATION MODE AND
IGNORES THE DEFINE BOARDS
ORIENTATION ON DEF.H**

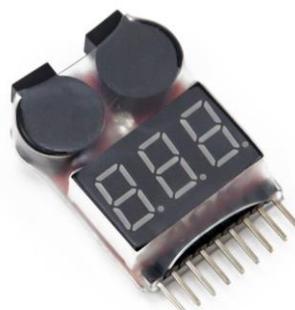
SENSORS IMU ORIENTATION IS IMPORTANT SEE TO IT THE ACC GYRO AND MAG ALL COMPLIMENT EACH OTHER



BATTERY CARE – ONLY CHARGE AT 1C OR THE RECOMMENDED THE CHARGE RATE ON THE LABEL

USE BALANCE CHARGER TO SET THE CURRENT IN THE SAMPLE IS 5A

**BATTERY STORAGE MODE IS 3.8V PER CELL
BATTERY DISCHARGE IS 3.6V PER CELL
BATTERY FULL CHARGE IS 4.2V PER CELL**

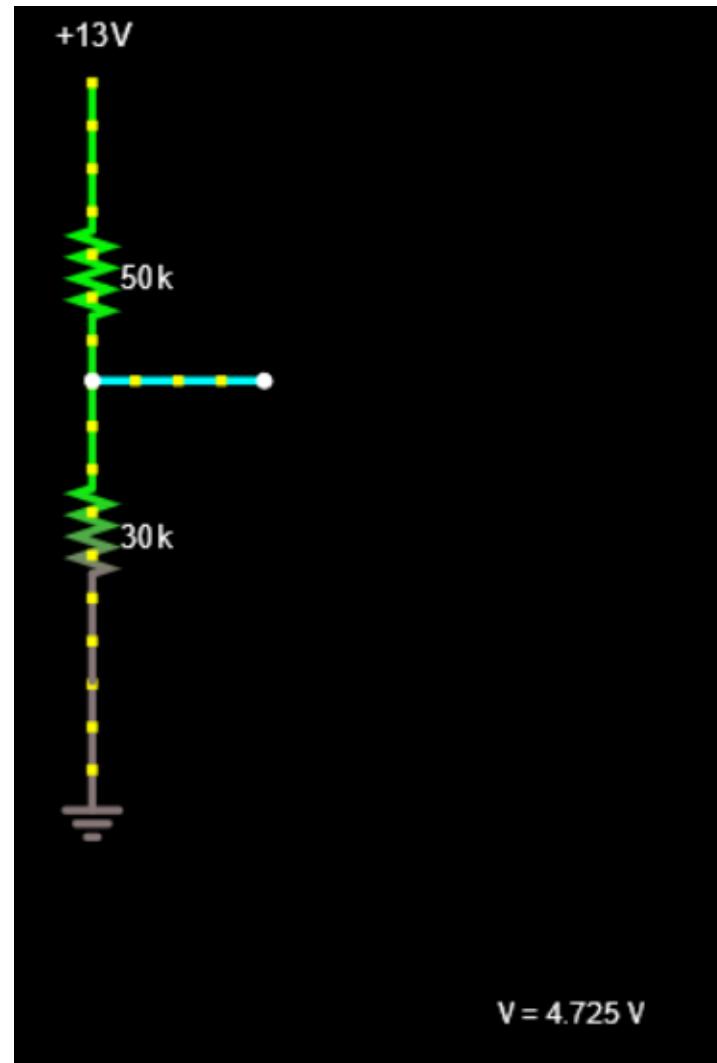
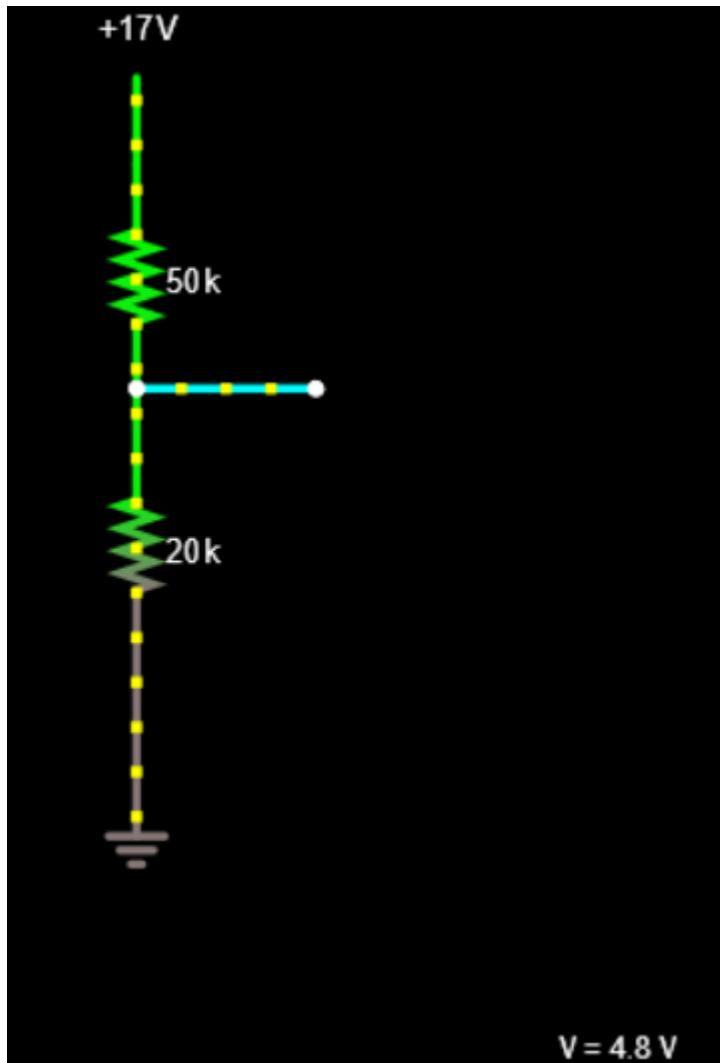


ITS RECOMMEND TO USE THE VOLTAGE ALARM TO MONITOR THE BATTERY VOLTAGE WHILE IN USE

BATTERY

4S 16.8V

3S 12.6V



VOLTAGE READING

HOW MUCH POWER IN YOUR
BATTERY

VOLTAGE DIVIDER

THIS ALLOWS THE 3V-5V TO BE
INPUTTED TO THE Ao ANALOG
PIN OF THE ARDUINO TO READ
THE BATTERY VOLTAGE

SWITCH THE VBAT JUMPER
ACCORDINGLY TO THE BATTER
CELLS YOUR USING

3S OR 4S

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp ▾ Sens

```
/*
 **** battery_voltage_monitoring ****
 ****

 /* for V BAT monitoring
 after the resistor divisor we should get [0V;5V]->[0;1023] on analog V_BATPIN
 with R1=33k and R2=51k
 vbat = [0;1023]*16/VBATSCALE
 must be associated with #define BUZZER ! */

 // #define VBAT // uncomment this line to activate the vbat code
 #define VBATSCALE 131 // (*) (**) change this value if readed Battery voltage is different than real voltage
 #define VBATNOMINAL 126 // 12,6V full battery nominal voltage - only used for lcd.telemetry
 #define VBATLEVEL_WARN1 107 // (*) (**) 10,7V
 #define VBATLEVEL_WARN2 99 // (*) (**) 9,9V
 #define VBATLEVEL_CRIT 93 // (*) (**) 9,3V - critical condition: if vbat ever goes below this value, permanent alarm is triggered
 #define NO_VBAT 16 // Avoid beeping without any battery
 #define VBAT_OFFSET 18 // offset in 0.1Volts, gets added to voltage value - useful for zener diodes

 /* for V BAT monitoring of individual cells
 * enable both VBAT and VBAT_CELLS
 */
 // #define VBAT_CELLS
 #define VBAT_CELLS_NUM 0 // set this to the number of cells you monitor via analog pins
 #define VBAT_CELLS_PINS {A0, A1, A2, A3, A4, A5} // set this to the sequence of analog pins
 #define VBAT_CELLS_OFFSETS {0, 50, 83, 121, 149, 177} // in 0.1 volts, gets added to voltage value - useful for zener diodes
 #define VBAT_CELLS_DIVS { 75, 122, 98, 18, 30, 37 } // divisor for proportional part according to resistors - larger value here gives smaller voltage

 **** powermeter (battery capacity monitoring) ****

 892 Arduino/Genuino Uno on COM41 10:24 PM 13/02/2020 ENG
```

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp

```
***** Buzzer Pin *****  
/* this moves the Buzzer pin from TXO to D8 for use with ppm sum or spectrum sat. RX (not needed if A32U4ALLPINS is active) */  
//#define D8BUZZER  
  
***** Promicro version related *****  
/* Inverted status LED for Promicro ver 10 */  
//#define PROMICRO10  
  
***** override default pin assignments *****  
  
/* only enable any of this if you must change the default pin assignment, e.g. your board does not have a specific pin */  
/* you may need to change PINx and PORTx plus #shift according to the desired pin! */  
#define OVERRIDE_V_BATPIN A0 // instead of A3 // Analog PIN 3  
  
##define OVERRIDE_PSENSORPIN A1 // instead of A2 // Analog PIN 2  
  
##define OVERRIDE_LEDPIN_PINMODE pinMode (A1, OUTPUT); // use A1 instead of d13  
##define OVERRIDE_LEDPIN_TOGGLE PINC |= 1<<1; // PINB |= 1<<5; //switch LEDPIN state (digital PIN 13)  
##define OVERRIDE_LEDPIN_OFF PORTC &= ~(1<<1); // PORTB &= ~(1<<5);  
##define OVERRIDE_LEDPIN_ON PORTC |= 1<<1; // was PORTB |= (1<<5);  
  
##define OVERRIDE_BUZZERPIN_PINMODE pinMode (A2, OUTPUT); // use A2 instead of d8  
##define OVERRIDE_BUZZERPIN_ON PORTC |= 1<<2 //PORTB |= 1;  
##define OVERRIDE_BUZZERPIN_OFF PORTC &= ~(1<<2); //PORTB &= ~1;  
  
*****  
Done Saving.  
479 Arduino/Genuino Uno on COM41 9:30 AM 21/02/2020 ENG
```



Main Mix Table

AGAVBoat2 - Output.cpp | Arduino 1.8.2

File Edit Sketch Tools Help

AGAVBoat2 Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp

```
// get servo middle point from Config or from RC-Data
int16_t get_middle(uint8_t nr) {
    return (conf.servoConf[nr].middle < RC_CHANS) ? rcData[conf.servoConf[nr].middle] : conf.servoConf[nr].middle;
}

// int8_t servodir(uint8_t n, uint8_t b) { return ((conf.servoConf[n].rate & b) ? -1 : 1); }

void mixTable() {
    int16_t maxMotor;
    uint8_t i;
    #if defined(DYNBALANCE)
        return;
    #endif
#define PIDMIX(X,Y,Z) rcCommand[THROTTLE] + axisPID[ROLL]*X + axisPID[PITCH]*Y + YAW_DIRECTION * axisPID[YAW]*Z
#define SERVODIR(n,b) ((conf.servoConf[n].rate & b) ? -1 : 1)

    //***** main Mix Table *****
    #if defined( MY_PRIVATE_MIXING )
        #include MY_PRIVATE_MIXING
    #elif defined( BI )
        motor[0] = PIDMIX(+1, 0, 0); //LEFT
        motor[1] = PIDMIX(-1, 0, 0); //RIGHT
        //Steering Drive
        //servo[4] = (SERVODIR(4,1) * axisPID[YAW]) + get_middle(4); //Steering Wheel
        //servo[5] = (SERVODIR(5,1) * axisPID[PITCH]) + get_middle(5); //Throttle
        //Differential Drive
        servo[4] = (SERVODIR(4,2) * axisPID[YAW]) + (SERVODIR(4,1) * axisPID[PITCH]) + get_middle(4); //LEFT
        servo[5] = (SERVODIR(5,2) * axisPID[YAW]) + (SERVODIR(5,1) * axisPID[PITCH]) + get_middle(5); //RIGHT
    #elif defined( TRI )
        motor[0] = PIDMIX( 0,+4/3, 0); //REAR
        motor[1] = PIDMIX(-1,-2/3, 0); //RIGHT
        motor[2] = PIDMIX(+1,-2/3, 0); //LEFT
        servo[5] = (SERVODIR(5, 1) * axisPID[YAW]) + get_middle(5); //REAR
    #endif
}

Done uploading.
avrduke done. Thank you.
```

1138-1146

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM3

Links ENG 7:31 PM 27/02/2022 2

Where you define your vehicle type
Differential Drive
Steering Drive

SERIAL COM

CONFIG.H



SERIAL BAUD RATE

WILL DEPEND ON WHAT BAUD
YOUR TELEMETRY MODULE ARE
SET INTO

SERIAL 0,1,3 IS RESERVE FOR TELEMETRY

115200 FOR BLUETOOTH HC-05

38400 FOR XBEE RADIO

SERIAL 2 IS RESERVE FOR GPS

NMEA Baud 57600

The screenshot shows the Arduino IDE interface with the file "MultiWii - config.h" open. The code is as follows:

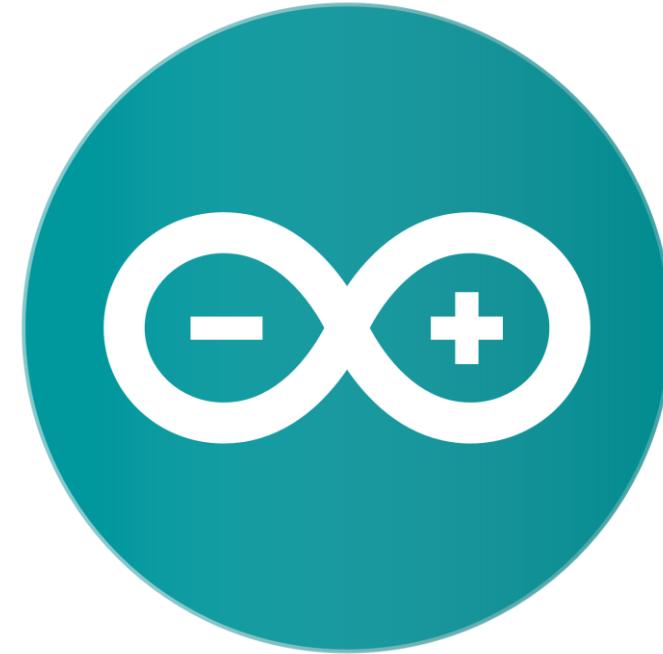
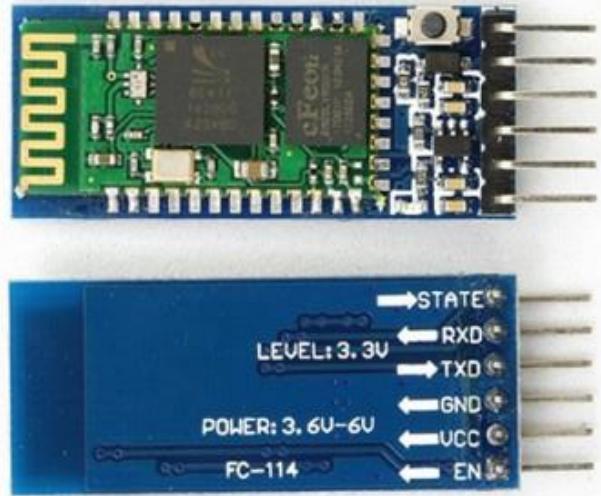
```
MultiWii - config.h | Arduino 1.8.2
File Edit Sketch Tools Help
MultiVii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp Sensors.h

/*
***** SECTION 5 - ALTERNATE SETUP *****
*****
***** Serial com speed *****
/* This is the speed of the serial interfaces */
#define SERIAL0_COM_SPEED 38400
#define SERIAL0_COM_SPEED 57600
#define SERIAL0_COM_SPEED 115200
#define SERIAL1_COM_SPEED 115200
#define SERIAL2_COM_SPEED 115200
#define SERIAL3_COM_SPEED 115200

/* when there is an error on I2C bus, we neutralize the values during a short time. expressed in microseconds
it is relevant only for a conf with at least a WMP */
#define NEUTRALIZE_DELAY 100000

/*
***** Gyro filters *****
*****
***** Lowpass filter for some gyros *****
/* ITG3200 & ITG3205 Low pass filter setting. In case you cannot eliminate all vibrations to the Gyro, you can try
to decrease the LPF frequency, only one step per try. As soon as twitching gone, stick with that setting.
It will not help on feedback wobbles, so change only when copter is randomly twitching and all dampening and
balancing options ran out. Uncomment only one option!
IMPORTANT! Change low pass filter setting changes PID behaviour, so retune your PID's after changing LPF.

498
Arduino/Genuino Uno on COM4!
```



BLUETOOTH

Bluetooth setup with the USB TTL and Arduino IDE

Arduino IDE>Tools>Serial Monitor (Push Button Before Connecting the USB) Set (Baud 38400) (Both NL & CR)

AT : check the connection

AT+VERSION : Check Version

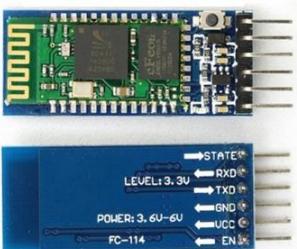
HC-05 (Recommended)

AT+NAME=Change name

AT+PSWD=1234 (Version 2)

AT+PSWD="1234" (Version 3)

AT+UART=115200,1,0



HC-05 (Recommended)

MultiWii | Arduino 1.8.5

File Edit Sketch Tools Help

Auto Format Ctrl+T

Archive Sketch

Fix Encoding & Reload

Serial Monitor Ctrl+Shift+M

Serial Plotter Ctrl+Shift+L

WiFi101 Firmware Updater

ArduBlock

Board: "Arduino/Genuino Uno"

Port

Get Board Info

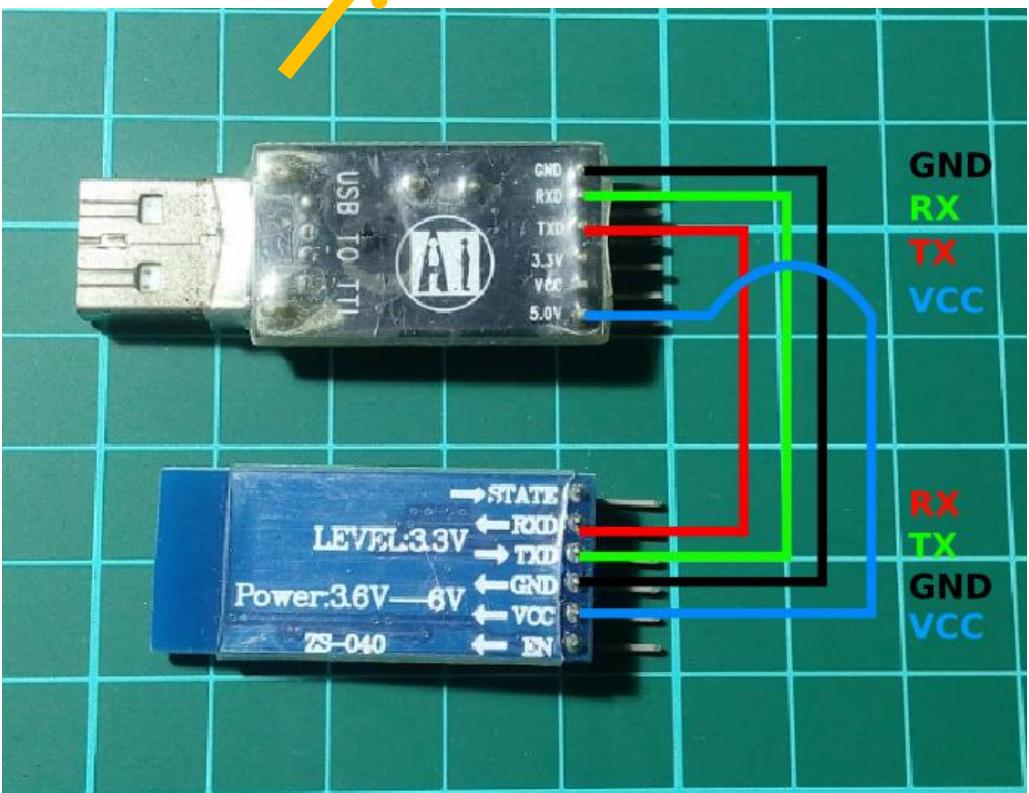
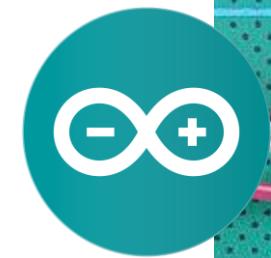
Programmer: "AVRISP mkII"

Burn Bootloader

```
/*
 * Welcome
 *
 * If you see this message, it means you are using the Arduino IDE. That is ok.
 * To get started with your copter, you must switch to the tab named 'config.h'.
 * Maybe there is a configuration file for your board or sensors and optionally various features.
 * For more information, please visit the MultiWii wiki at
 * /wiki/index.php?title=Main_Page
 *
 * Have fun!
 */

```





Bluetooth setup with the USB TTL and Arduino IDE

Arduino IDE>Tools>Serial Monitor (hold the Button while Plugging the USB) to go programming mode Set (Baud 38400) (Both NL & CR)

Hold Press when sending AT command (Version 5)

AT : check the connection

AT+VERSION : Check Version

HC-05 (Recommended)

AT+NAME=Change name (Synerduino)

AT+PSWD=1234 (Version 2)

AT+PSWD="1234" (Version 3) (Possibly works on Version 5)

AT+UART=115200,1,0 **(115200 FOR BLUETOOTH)**

HC-06

AT+NAME: Change name

AT+PIN: change pin, xxxx is the pin, again, no space.

AT+BAUDX, where X=1 to 9

1 set to 1200bps, 2 set to 2400bps, 3 set to 4800bps

4 set to 9600bps (Default)

5 set to 19200bps, 6 set to 38400bps, 7 set to 57600bps

8 set to 115200bps

HM-10 Bluetooth

Setup with FTDI + Arduino Serial Monitor + AT Command

AT+NAME? (Query name)

AT+ADDR? ((Query Mac address)

First you will need to Query the native MAC address using AT Command **AT+ADDR?**

You will get something like this `20C38FF61DA1`, each BLE has a unique MAC address.

Use **AT+CON[param1]** and **AT+ROLE[param1]** to pair to another device.

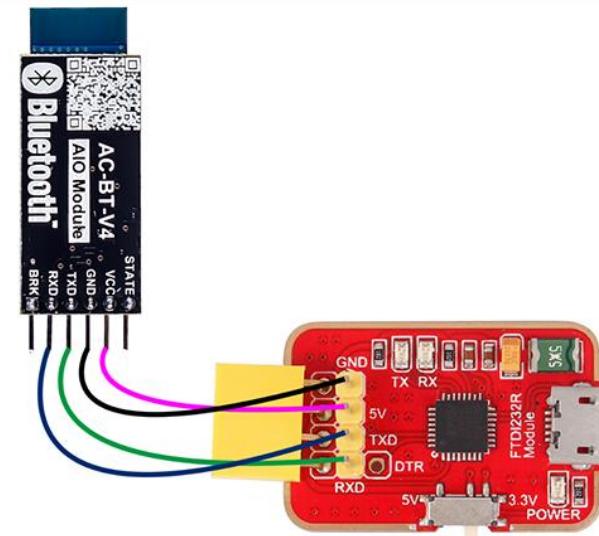
Example

BLE A has Mac Address `11C11FF11DA1`, I used **AT+ADDR?** to figure it out BLE B has Mac Address `22C22FF22DA2`, I used **AT+ADDR?** to figure it out

Send **AT+CON22C22FF22DA2** to BLE A Send **AT+CON11C11FF11DA1** to BLE B (Send the B address to A, A address to B)

Send **AT+ROLE0** to BLE ASend **AT+ROLE1** to BLE B (Doesn't matter which one)

Now it's ready to use on you ATMEGA 328P, Arduino or Attiny. **The red light will stay solid after the connection has been made on both BLE. This should take less than a second.**



HM-10 (Original)

AT (Check if new configuration is working)

AT+NAME (Query name)

AT+ADDR (Query Mac address)

AT+BAUD (Query Baud)

AT+PASS (Query current Pincode)

AT+PIN (Query current Pincode on some BL module)

AT+TYPE (Query authentication mode)

AT+ROLE (Query Peripheral (Slave) or Central (Master) mode)

AT+TYPE

0:Not need PIN Code

1:Auth not need PIN

2:Auth with PIN

3:Auth and bond

AT+BAUD

0 – 9600:

1 – 19200

2 – 38400

3 – 57600 (Some BL its 4800)

4 – 115200

5 – 4800

6 – 2400

7 – 1200

8 – 230400 (Some BL its 115200)

AT+NAME ArduinoDrone

AT+BAUD₄ set baud to 115200 (we want this for high speed)

AT+BAUD₈ set baud to 115200 (on some BL module)

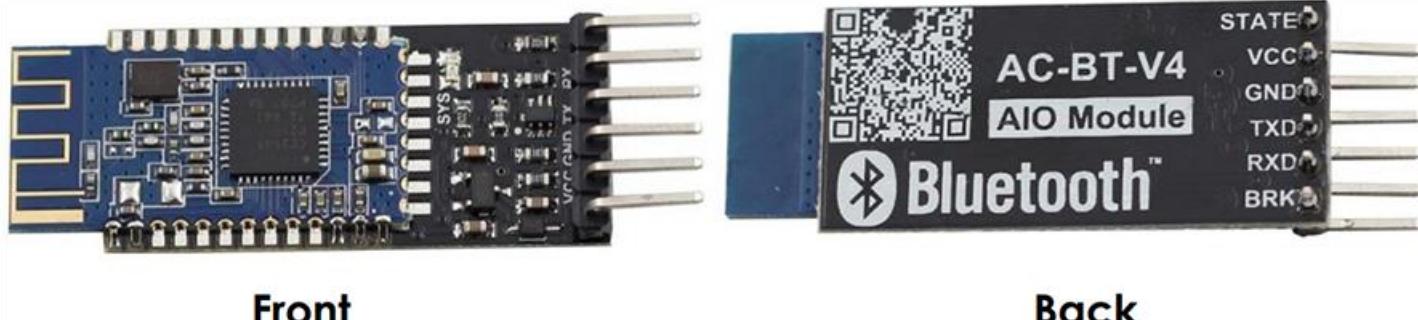
AT+PASS₁₂₃₄₅₆ Set password to 123456

AT+PIN₁₂₃₄₅₆ Set password to 123456 (on some BL module)

AT+ROLE

0 = Slave or Peripheral

1 = Master or Central.



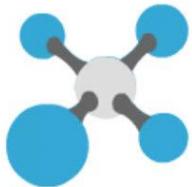
Front

Back

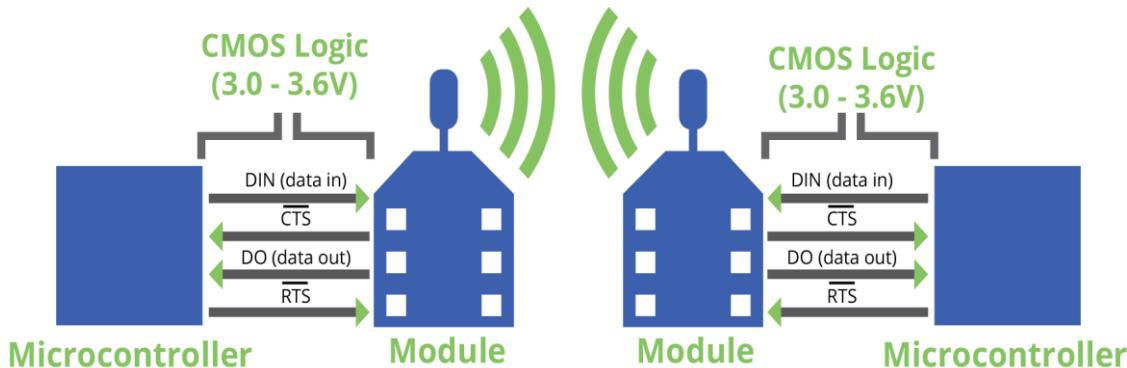
Note : there are several clones of this type in the market that can be very difficult to setup



XBEE RADIO



GROUND STATION ROUTER
38400 8/N/1/N - AT



AIRCRAFT COORDINATOR 38400
8/N/1/N - AT

Update firmware

Update the radio module firmware

Configure the firmware that will be flashed to the radio module.

Select the product family of your device, the new function set and the firmware version to flash:

Product family	Function set	Firmware version
XB24-B	ZigBee End Device Digital IO	22A7 (Newest)
XB24-SE	ZigBee End Device PH	22A0
XB24-ZB	ZigBee Router API	228C
	ZigBee Router AT	2270
	ZigBee Router AT (WALL RT)	2264
	ZigBee Router Sensor	2242
	ZigBee Router/End Device Analog IO	2241

Force the module to maintain its current configuration.

Select current

[View Release Notes](#)

[Update](#) [Cancel](#)

Update firmware

Update the radio module firmware

Configure the firmware that will be flashed to the radio module.

Select the product family of your device, the new function set and the firmware version to flash:

Product family	Function set	Firmware version
XB24-B	End Device - LTH	20A7 (Newest)
XB24-SE	ZigBee Coordinator API	20A0
XB24-ZB	ZigBee Coordinator AT	208C
	ZigBee End Device API	2070
	ZigBee End Device AT	2064
	ZigBee End Device Analog IO	2041
	ZigBee End Device Digital IO	2021

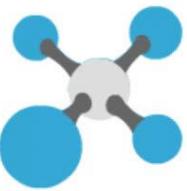
Force the module to maintain its current configuration.

Select current

[View Release Notes](#)

[Update](#) [Cancel](#)

GROUND STATION



XCTU

Radio Modules

Name: ZigBee Router AT
Function: ZigBee Router AT
Port: COM35 - 38400/8/N/1/N - AT
MAC: 0013A20040811A91

Radio Configuration [- 0013A20040811A91]

ID PAN ID: 1234

SC Scan Channels: FFFF Bitfield

SD Scan Duration: 3 exponent

ZS ZigBee Stack Profile: 0

NJ Node Join Time: FF x 1 sec

NW Network Watchdog Timeout: 0 x 1 minute

JV Channel Verification: Disabled [0]

JN Join Notification: Disabled [0]

OP Operating PAN ID: 1234

OI Operating 16-bit PAN ID: AD9F

CH Operating Channel: 14

NC Number of Remaining Children: C

Addressing

Change addressing settings

SH Serial Number High: 13A200

SL Serial Number Low: 40811A91

MY 16-bit Network Address: 7FA4

DH Destination Address High: 13A200

DL Destination Address Low: 40811A7F

NI Node Identifier:

NH Maximum Hops: 1E

BH Broadcast Radius: 0

AR Many-to-One Route Broadcast Time: FF x 10 sec

DD Device Type Identifier: 30000

NT Node Discovery Backoff: 3C x 100 ms

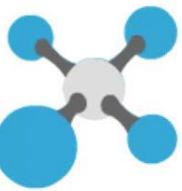
Checking for Radio Firmw... updates: (87%)

2:30 PM 07/04/2020

STM-5
94V-0 32 I/O
2010 DIGI INT'L
38069582-63 REV A

CE N151/21
XB24-Z7UIT-
004 RevG
0013A200
40811A7F
002811-10 50

GROUND STATION



XCTU

Radio Modules

Name: ZigBee Router AT
Function: ZigBee Router AT
Port: COM35 - 38400/8/N/1/N - AT
MAC: 0013A20040811A91

Radio Configuration [- 0013A20040811A91]

POWER ALIVE

Parameter

Security

Change security parameters

- EE Encryption Enable: Disabled [0]
- EO Encryption Options: Bitfield [0]
- KY Encryption Key: []

Serial Interfacing

Change modem interfacing options

- BD Baud Rate: 38400 [5]
- NB Parity: No Parity [0]
- SB Stop Bits: One stop bit [0]
- RO Packetization Timeout: 3 x character times
- D7 DIO7 Configuration: CTS flow control [1]
- D6 DIO6 Configuration: Disable [0]

AT Command Options

Change AT command mode behavior

- CT AT Command Mode Timeout: 64 x 100ms
- GT Guard Times: 3E8 x 1ms
- CC Command Sequence Character: 2B Recommended: 0x20-0x7F (ASCII)

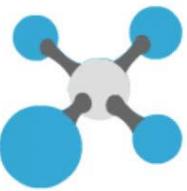
Sleep Modes

Configure low power options to support end device children

- SM Sleep Mode: No Sleep (Router) [0]
- SN Number of Cyclic Sleep Periods: 1
- SO Sleep Options: 0
- SP Cyclic Sleep Period: 20 x 10 ms
- ST Time before Sleep: 1388 x 1 ms

Image of the XB24-Z7UIT module:

AIRCRAFT



XCTU

Radio Modules

Name: ZigBee Coordinator AT
Function: ZigBee Coordinator AT
Port: COM36 - 38400/8/N/1/N - AT
MAC: 0013A20040811A7F

Radio Configuration [- 0013A20040811A7F]

Networking

Change networking settings

① ID PAN ID	1234			
① SC Scan Channels	FFFF	Bitfield		
① SD Scan Duration	3	exponent		
① ZS ZigBee Stack Profile	0			
① NJ Node Join Time	FF	x 1 sec		
① OP Operating PAN ID	1234			
① OI Operating 16-bit PAN ID	AD9F			
① CH Operating Channel	14			
① NC Number of Remaining Children	A			

Addressing

Change addressing settings

① SH Serial Number High	13A200		
① SL Serial Number Low	40811A7F		
① MY 16-bit Network Address	0		
① DH Destination Address High	13A200		
① DL Destination Address Low	40811A91		
① NI Node Identifier			
① NH Maximum Hops	1E		
① BH Broadcast Radius	0		
① AR Many-to-One Route Broadcast Time	FF x 10 sec		
① DD Device Type Identifier	30000		
① NT Node Discovery Backoff	3C x 100 ms		
① NO Node Discovery Options	0		
① NP Maximum Number of Transmission Bytes	54		

SIK SERIAL RADIO

38400 OR 57600 FOR SIK
RADIO DEPENDING IF USES
433MHZ OR 900MHZ

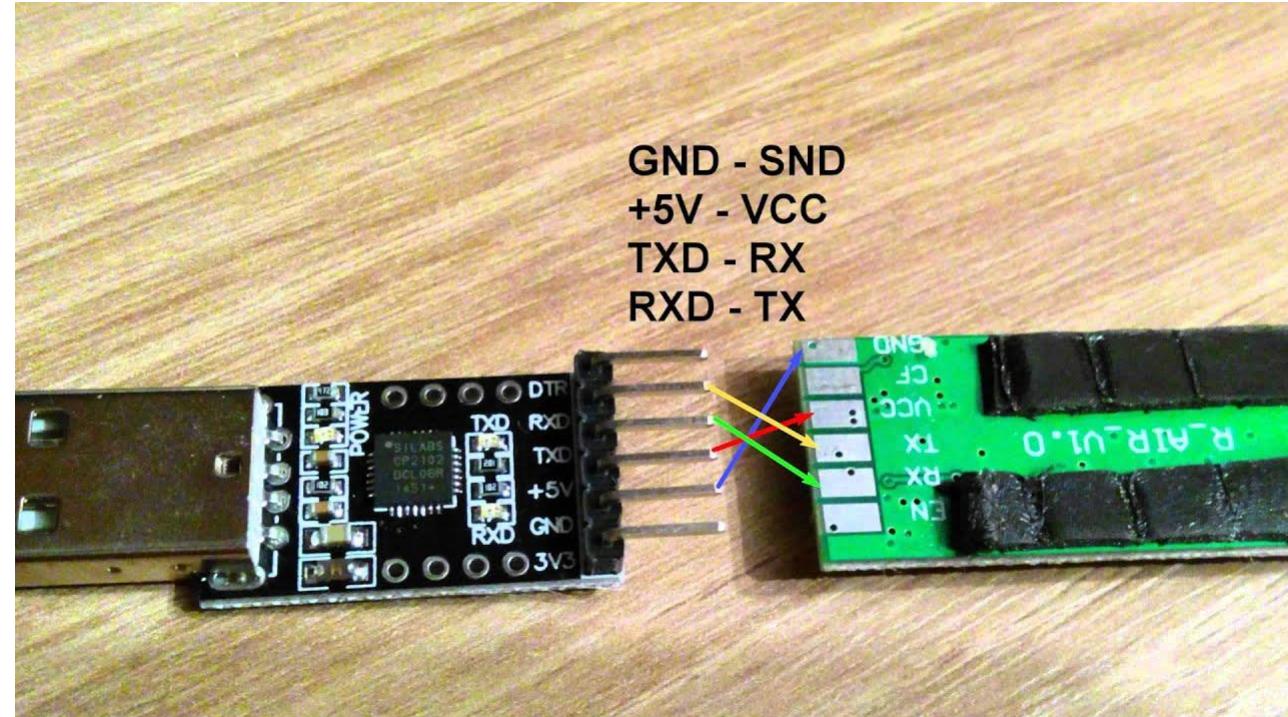


RadioTelemetry Air Module



RadioTelemetry Ground Module

Again to setup you require an USB-TTL module to connect to the serial port to configure both the module how ever most likely you only need to do this for the vehicle unit as the ground unit has an USB build into



GND - SND
+5V - VCC
TXD - RX
RXD - TX

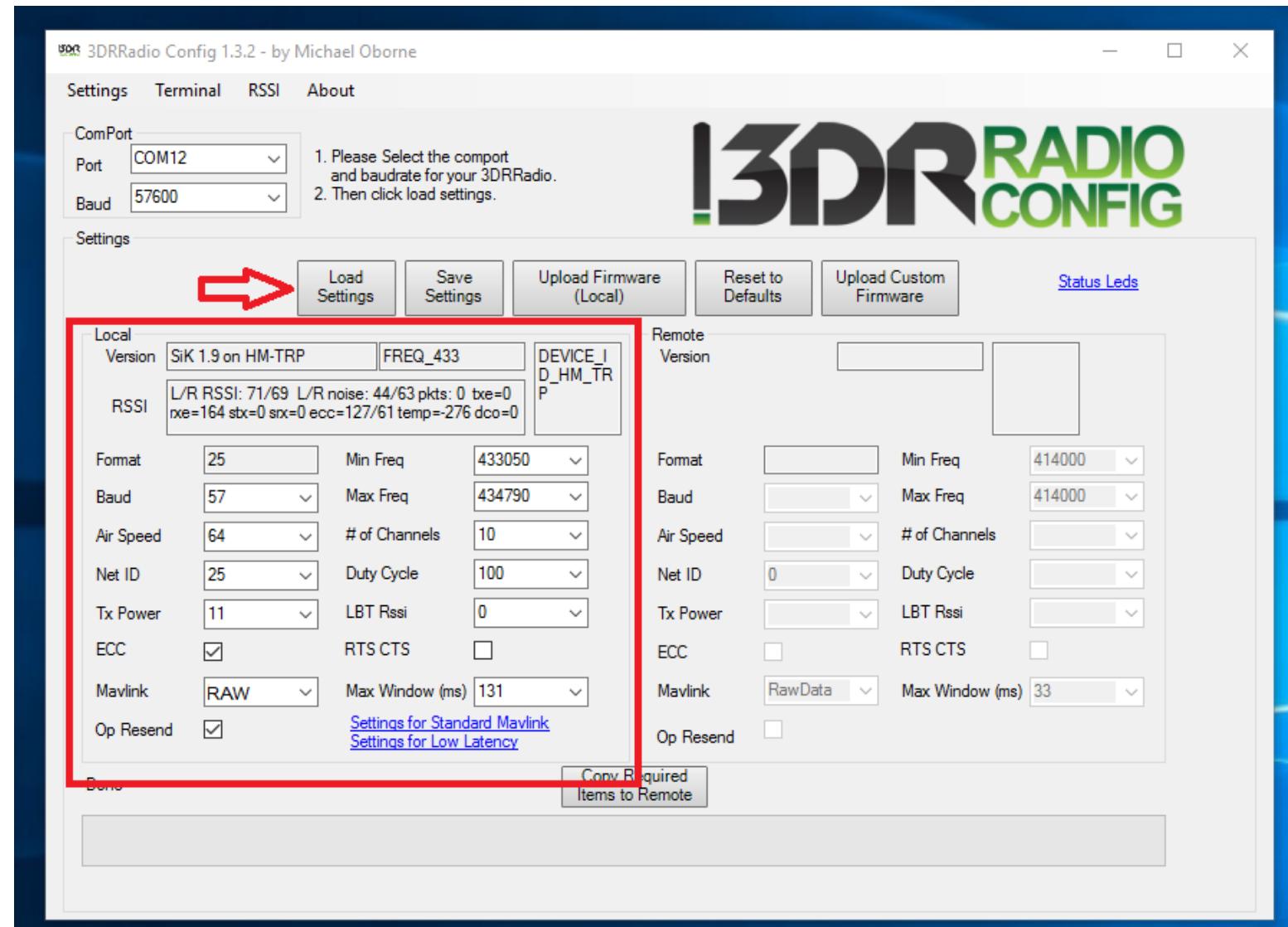
Manually configuring the telemetry kit for Synerduino uses the 3DR radio Config

<http://vps.oborne.me/3drradioconfig.zip>

Also available in the synerduino page

Both Vehicle and Ground station unit must have similar in the following

- Versions
- Frequency
- Baud (38400 or 57600 ensure)
- Airspeed
- Net ID (in cases you need to assign multiple drones each having their own ID)
- Tx power
- Mavlink (RAW –Synerduino uses Format)



GPS

Beitian



UBLOX Protocol

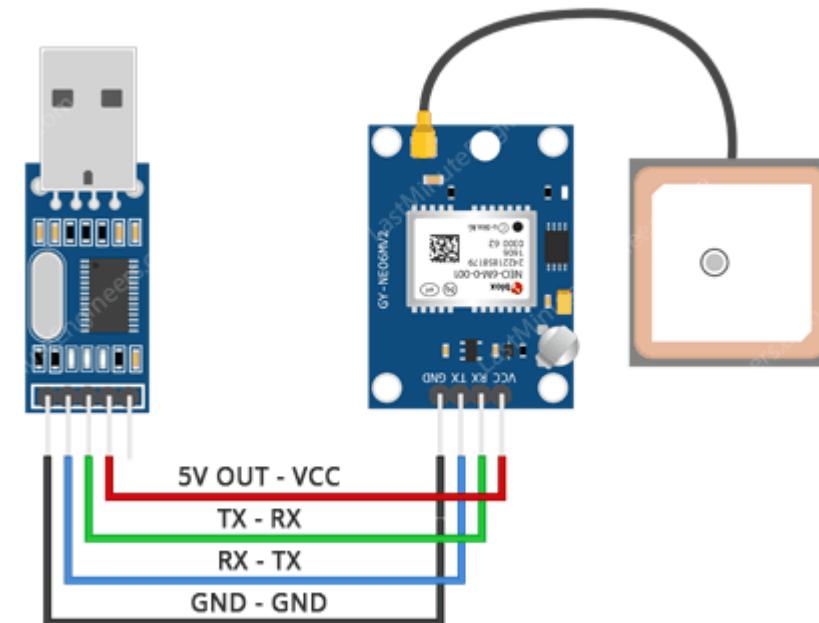
ublox



NMEA Protocol

NOTE: GPS CONFIGURING ONLY WORKS WHEN GPS MODULE COMES WITH EEPROM OR FLASH MEMORY. AS YOUR SELLER IF IT COMES WITH THOSE FUNCTIONS

GPS CONFIGURING



U BLOX NEO 6

PLUG IN TO SERIAL TX 2 RX 2

USB TTL TO PROGRAM THE GPS

THIS GOES SAME ON THE DRONE SHIELD

GPS CONFIGURING



**4.VCC
3.RX
2.TX
1.GND**

PIN	PIN Name	I/O	Description
1	GND	G	Ground
2	TX	O	Serial Data Output.
3	RX	I	Serial Data Input.
4	VCC	I	DC 3.0V - 5.5V supply input,Typical: 5.0V

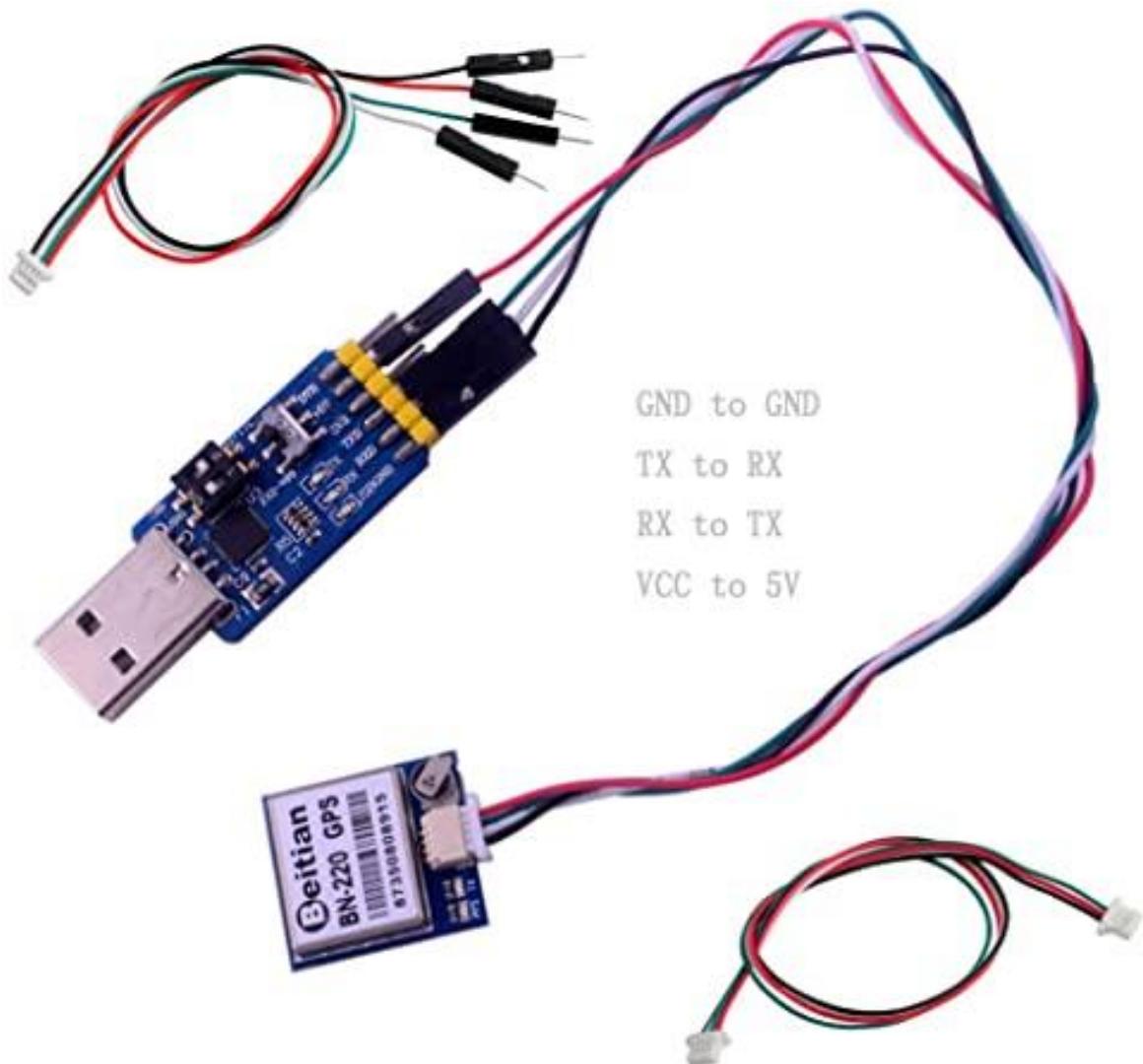


Pin Description:

PIN	PIN Name	I/O	Description
1	SDA	O	Compass SDA
2	GND	G	Ground
3	TX	O	Serial Data Output.
4	RX	I	Serial Data input.
5	VCC	I	3.0V~ 5.5V supply input,Typical: 5.0V
6	SCL	I	Compass SCL

GPS ONLY MODELS & GPS WITH COMPASS MODEL (HMC5883)

GPS CONFIGURING



BEITIAN UBLOX

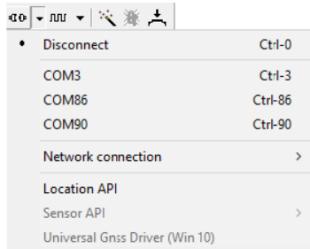
PLUG IN TO SERIAL TX 2 RX 2

USB TTL TO PROGRAM THE GPS

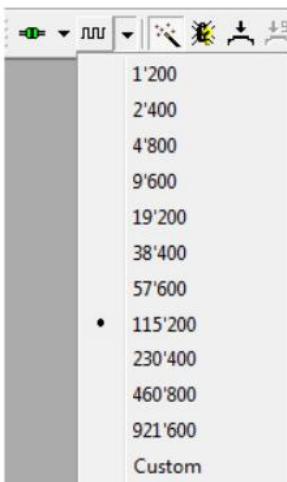
THIS GOES SAME ON THE DRONE SHIELD



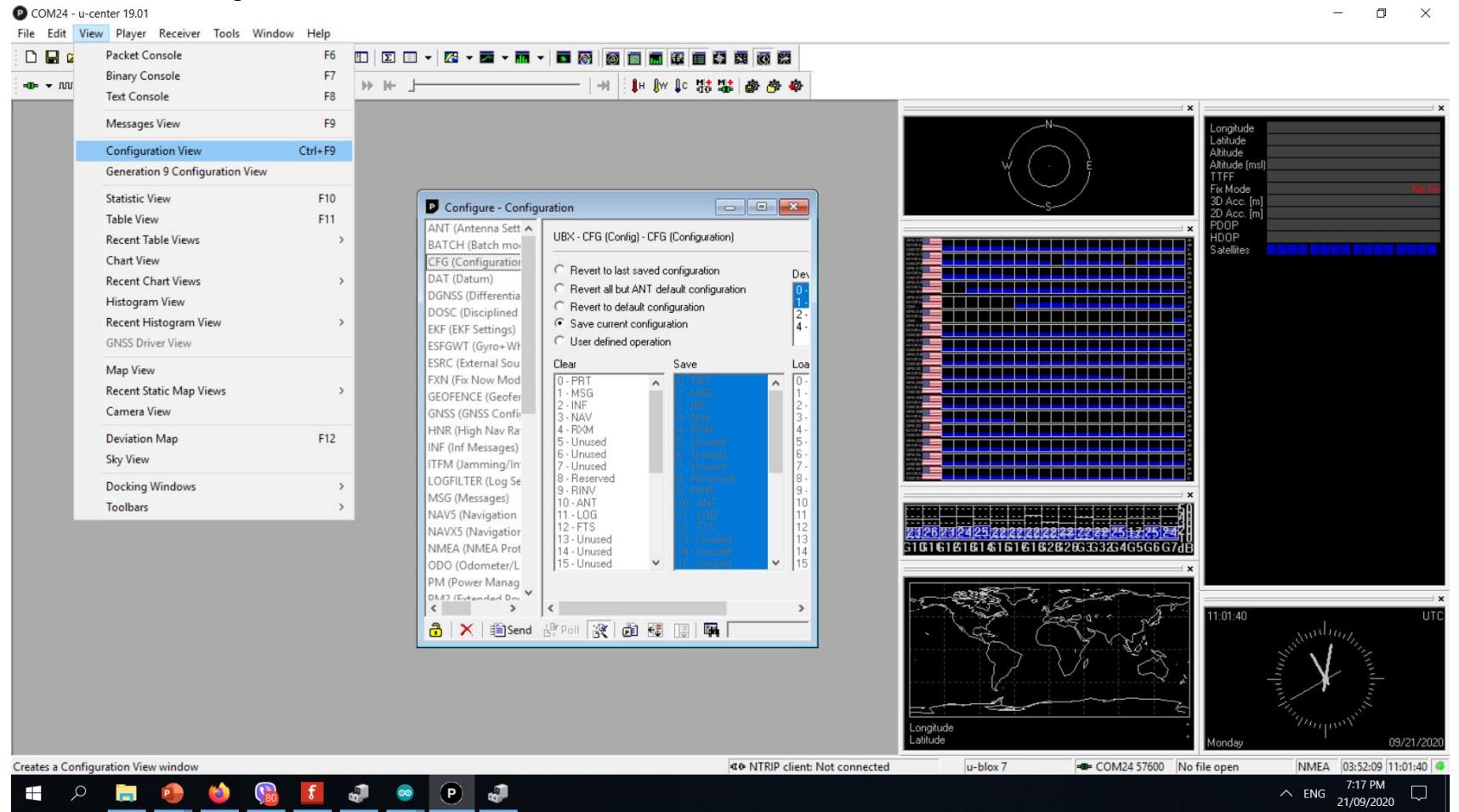
Connect to the device. Select the COM port your GPS and USBTTL is connected to



Connect to the baud rate your GPS is set to Default (9600) Setup we wanted (57600)



Go to View> Configuration View

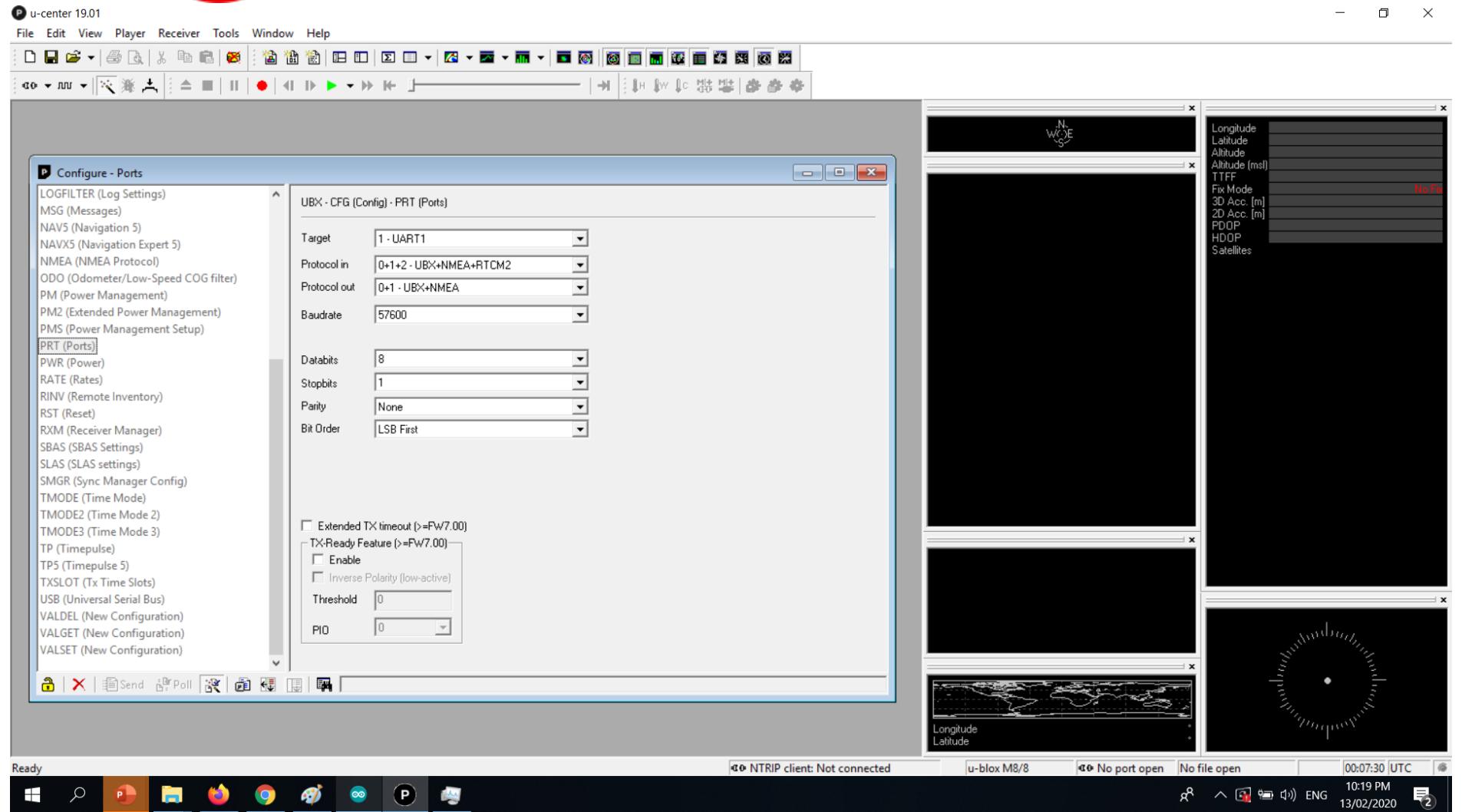


U CENTER



ublox

1. Connect to the device.
2. Open View / Messages View
3. Select UBX-CFG-PRT.
4. Poll the current configuration from the receiver Change the setting to the desired baud rate.
(57600)
5. Click the lock icon on the lower left and Send the message to the receiver

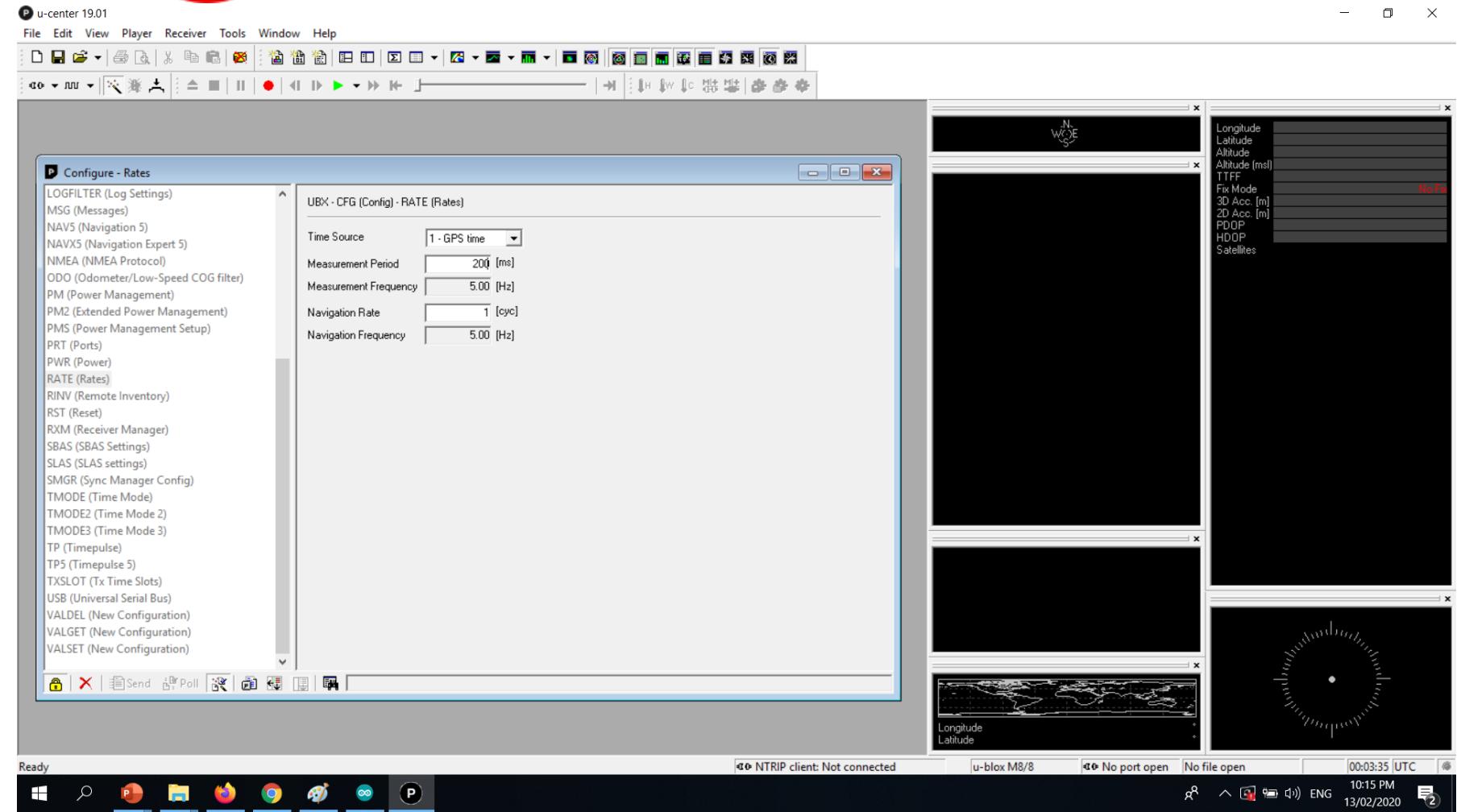




Rate zooms
Frequency 5hz

Note after updating
the settings click o
the Lock icon on the
left and click the
send button

Exit the
configuration
window this will
prompt you to the
save parameters .
Hit “Yes”





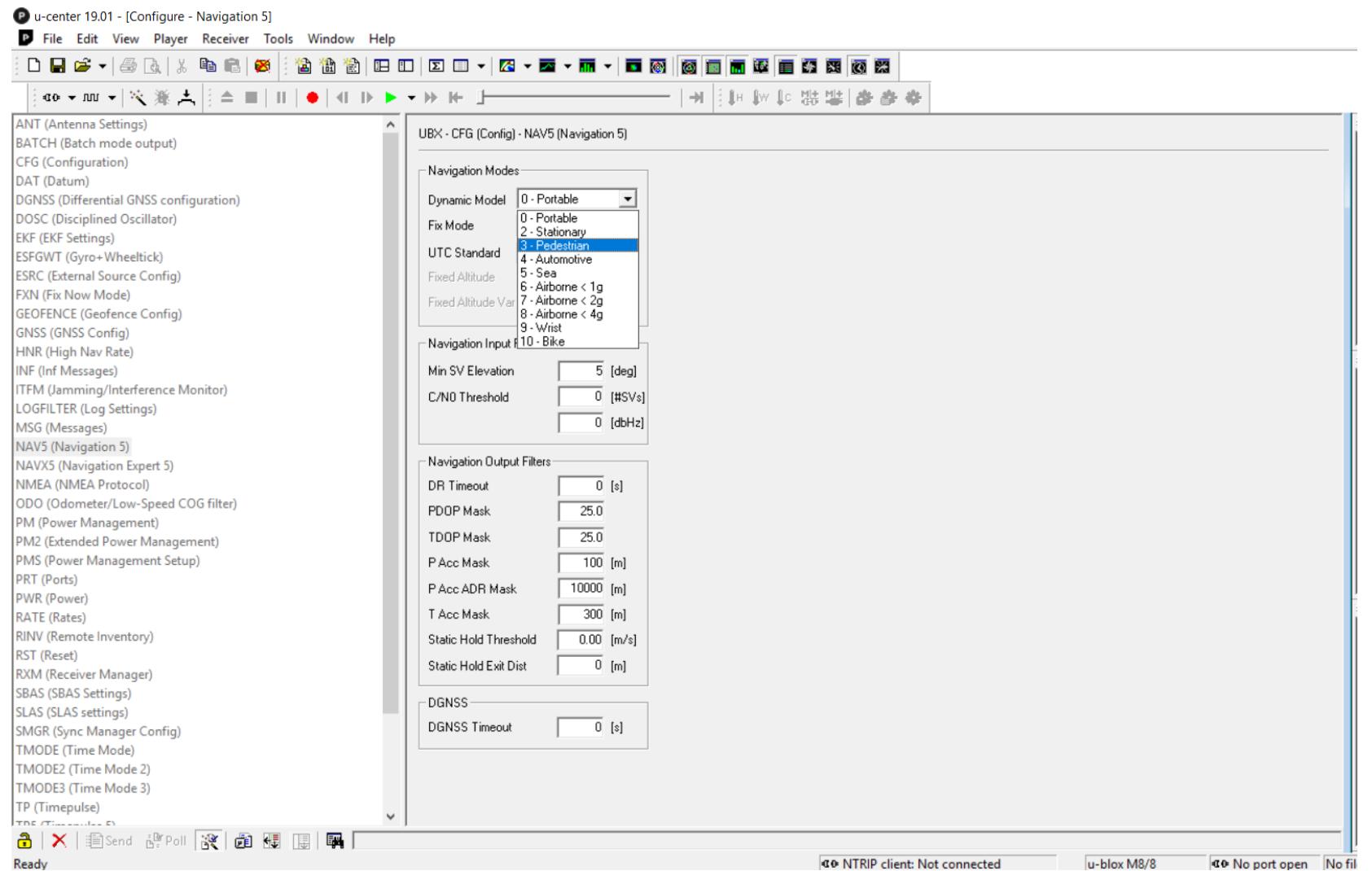
Nav5 (Navigation 5)

Dynamic Model

3. Pedestrian – better position hold

6-8 Airborne – ideal for Navigation , this reduces filtering

Hit Send icon to save



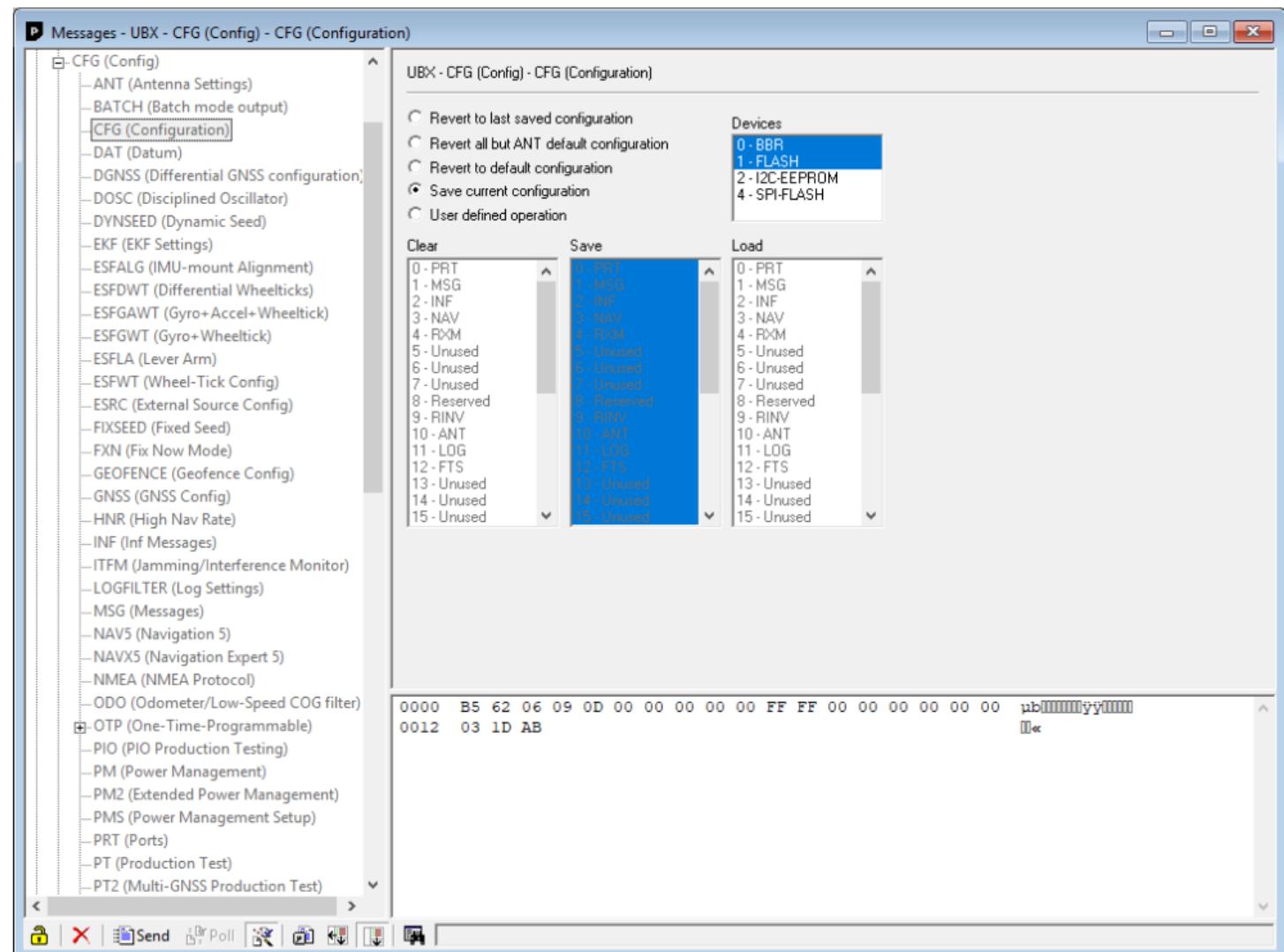
U CENTER



The logo for U CENTER features the word "U CENTER" in a bold, black, sans-serif font. To the right of the text is the ublox logo, which consists of a red circle containing a white lowercase 'i' followed by the word "ublox" in a bold, black, sans-serif font.

1. Connect to the device.
 2. Open View / Messages View
 3. Select UBX-CFG-CFG.
 4. Select “save current configuration”
 5. Send the message to the receiver
hit the Lock icon then send icon
in the bottom

**Disconnect /
Reconnect to the GPS
again using the 57600
baud to see if the
parameters are save**



GPS

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp Sensors.h

```
/*
 * introduce a deadband around the stick center
 * Must be greater than zero, comment if you dont want a deadband on roll, pitch and yaw */
#define DEADBAND 6

/*
 * ENable this for using GPS simulator (NMEA only) */
#define GPS_SIMULATOR

/*
 * GPS using a SERIAL port
 * if enabled, define here the Arduino Serial port number and the UART speed
 * note: only the RX PIN is used in case of NMEA mode, the GPS is not configured by multiwii
 * in NMEA mode the GPS must be configured to output GGA and RMC NMEA sentences (which is generally the default conf for most GPS devices)
 * at least 5Hz update rate. uncomment the first line to select the GPS serial port of the arduino */

#define GPS_SERIAL 2 // should be 2 for flyduino v2. It's the serial port number on arduino MEGA
// must be 0 for PRO_MINI (ex GPS_PRO_MINI)
// note: Now a GPS can share MSP on the same port. The only constrain is to not use it simultaneously, and use the same port speed.

// avoid using 115200 baud because with 16MHz arduino the 115200 baudrate have more than 2% speed error (57600 have 0.8% error)
#define GPS_BAUD 38400 // ublox 8 new standard
#define GPS_BAUD 9600
#define GPS_BAUD 57600 // GPS_BAUD will override SERIALx_COM_SPEED for the selected port (my ublox 6)
#define GPS_BAUD 115200

Done Saving.
```

676 Arduino/Genuino Uno on COM41 4:05 PM 20/02/2020

Set GPS BAUD to 57600 as configured to the GPS module

GPS

CONFIG.H



SynerduinoKwad3 - config.h | Arduino 1.8.5

File Edit Sketch Tools Help

SynerduinoKwad3 Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sens.cpp

```
// avoid using 115200 baud because with 16MHz arduino the 115200 baudrate have more than 2% speed error (57600 have 0.8% error)
//#define GPS_BAUD 9600 // GPS default
//#define GPS_BAUD 38400
#define GPS_BAUD 57600 // GPS_BAUD will override SERIALX_COM_SPEED for the selected port
//#define GPS_BAUD 115200

/* 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 !!! */

SEE UCENTER https://www.u-blox.com/en/product/u-center */

#ifndef NMEA //for Ublox NMEA GPS - NMEA Protocol
#define UBLOX //for Beitian GPS - UBX Protocol
#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

/* 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 */
/* all functionnalities allowed by SERIAL_GPS are now available for I2C_GPS: all relevant navigation computations are gathered in the main FC */

#define I2C_GPS
```

694 - 699

Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM25

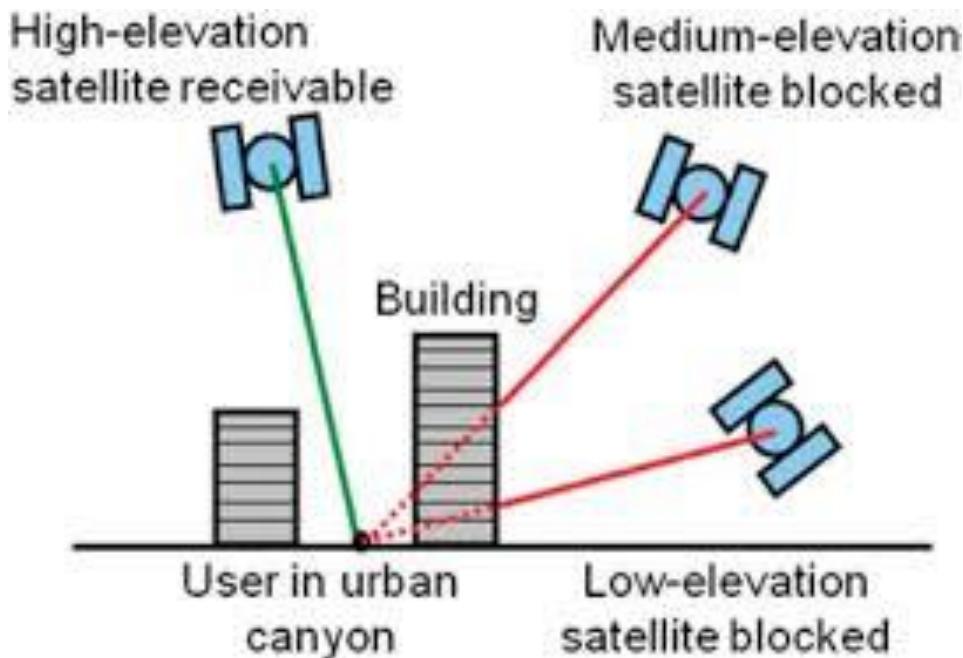
^ ENG 2:59 PM 30/09/2021

Depending on the GPS Version

Older formats uses
NMEA protocol

Newer GPS uses
UBLOX UBX Protocol

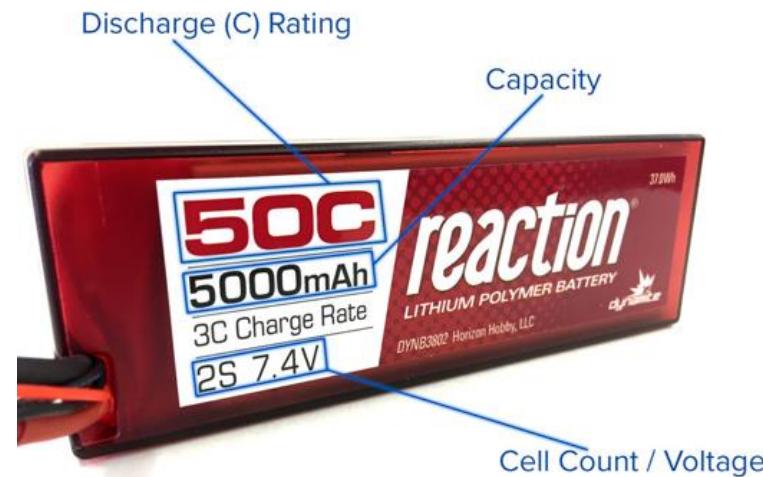
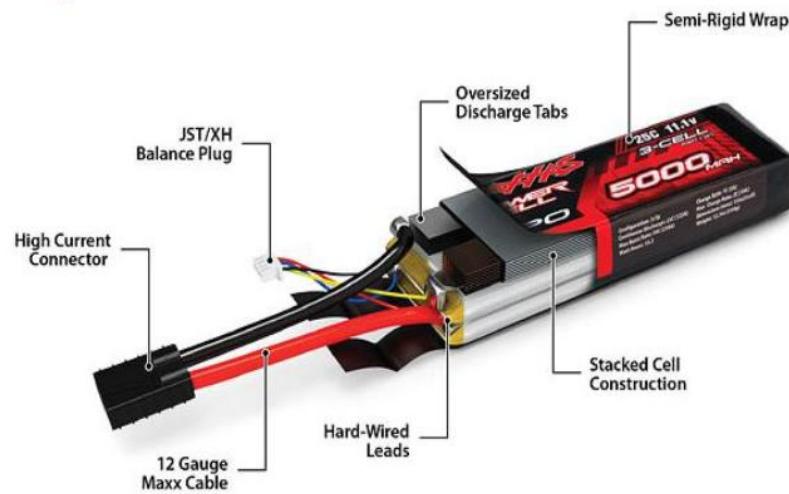
GPS



Note : GPS require a clear open area to get a proper fix and accuracy minimum 7 satellites but 10+ are Ideal

Flying next to a building can distort satellite signal deteriorating accuracy

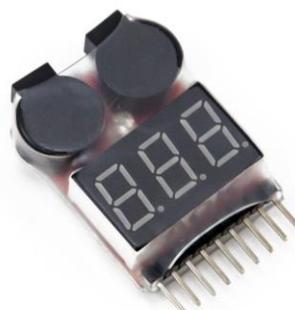
Which in this case its better to not use GPS modes and fly Manual



BATTERY CARE – ONLY CHARGE AT 1C OR THE RECOMMENDED THE CHARGE RATE ON THE LABEL

USE BALANCE CHARGER TO SET THE CURRENT IN THE SAMPLE IS 5A

**BATTERY STORAGE MODE IS 3.8V PER CELL
BATTERY DISCHARGE IS 3.6V PER CELL
BATTERY FULL CHARGE IS 4.2V PER CELL**

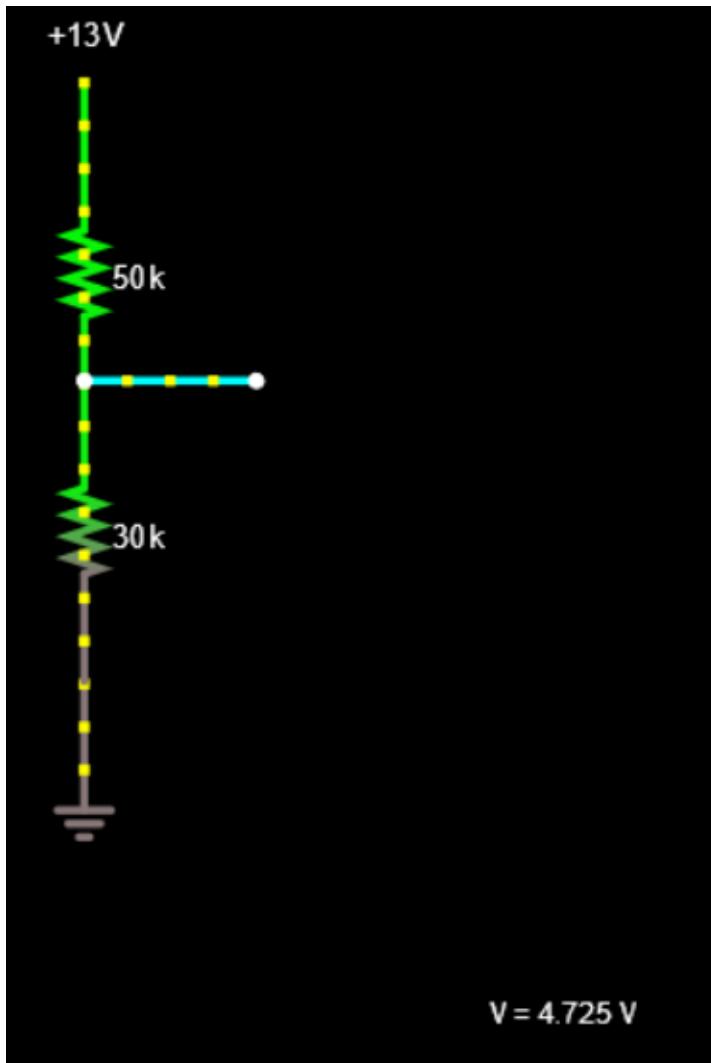


ITS RECOMMEND TO USE THE VOLTAGE ALARM TO MONITOR THE BATTERY VOLTAGE WHILE IN USE

BATTERY

4S 16.8V

3S 12.6V



VOLTAGE READING

HOW MUCH POWER IN YOUR
BATTERY

VOLTAGE DIVIDER

THIS ALLOWS THE 3V-5V TO BE
INPUTTED TO THE Ao ANALOG
PIN OF THE ARDUINO TO READ
THE BATTERY VOLTAGE

SWITCH THE VBAT JUMPER
ACCORDINGLY TO THE BATTER
CELLS YOUR USING

3S OR 4S

BATTERY MONITORING

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

Multivii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp ▾ Sens

```
/*
***** battery_voltage_monitoring *****
*****
/* for V BAT monitoring
   after the resistor divisor we should get [0V;5V]->[0;1023] on analog V_BATPIN
   with R1=33k and R2=51k
   vbat = [0;1023]*16/VBATSCALE
   must be associated with #define BUZZER ! */
#ifndef VBAT           // uncomment this line to activate the vbat code
#define VBATSCALE      131 // (*) (**) change this value if readed Battery voltage is different than real voltage
#define VBATNOMINAL    126 // 12,6V full battery nominal voltage - only used for lcd.telemetry
#define VBATLEVEL_WARN1 107 // (*) (**) 10,7V
#define VBATLEVEL_WARN2 99 // (*) (**) 9,9V
#define VBATLEVEL_CRIT 93 // (*) (**) 9,3V - critical condition: if vbat ever goes below this value, permanent alarm is triggered
#define NO_VBAT         16 // Avoid beeping without any battery
#define VBAT_OFFSET     18 // offset in 0.1Volts, gets added to voltage value - useful for zener diodes

/* for V BAT monitoring of individual cells
 * enable both VBAT and VBAT_CELLS
 */
#ifndef VBAT_CELLS
#define VBAT_CELLS_NUM 0 // set this to the number of cells you monitor via analog pins
#define VBAT_CELLS_PINS {A0, A1, A2, A3, A4, A5 } // set this to the sequence of analog pins
#define VBAT_CELLS_OFFSETS {0, 50, 83, 121, 149, 177 } // in 0.1 volts, gets added to voltage value - useful for zener diodes
#define VBAT_CELLS_DIVS { 75, 122, 98, 18, 30, 37 } // divisor for proportional part according to resistors - larger value here gives smaller voltage

***** powermeter (battery capacity monitoring) *****

```

892

Arduino/Genuino Uno on COM4!

10:24 PM 13/02/2020

The bottom of the screen shows a Windows taskbar with several icons: Start button, File Explorer, Microsoft Edge, Google Chrome, Paint 3D, Mail, and a red notification badge icon.

PIN ASSIGNMENTS

CONFIG.H



Multiwii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp sens

```
***** Buzzer Pin *****/
/* this moves the Buzzer pin from TXO to D8 for use with ppm sum or spectrum sat. RX (not needed if A32U4ALLPINS is active) */
#define D8BUZZER

***** Promicro version related *****/
/* Inverted status LED for Promicro ver 10 */
#define PROMICRO10

***** override default pin assignments *****/
/* only enable any of this if you must change the default pin assignment, e.g. your board does not have a specific pin */
/* you may need to change PINx and PORTx plus #shift according to the desired pin! */
#define OVERRIDE_V_BATPIN A0 // instead of A3 // Analog PIN 3

#define OVERRIDE_PSENSORPIN A1 // instead of A2 // Analog PIN 2

#define OVERRIDE_LEDPIN_PINMODE pinMode (A1, OUTPUT); // use A1 instead of d13
#define OVERRIDE_LEDPIN_TOGGLE PINC |= 1<<1; // PINB |= 1<<5; //switch LEDPIN state (digital PIN 13)
#define OVERRIDE_LEDPIN_OFF PORTC &= ~(1<<1); // PORTB &= ~(1<<5);
#define OVERRIDE_LEDPIN_ON PORTC |= 1<<1; // was PORTB |= (1<<5);

#define OVERRIDE_BUZZERPIN_PINMODE pinMode (A2, OUTPUT); // use A2 instead of d8
#define OVERRIDE_BUZZERPIN_ON PORTC |= 1<<2 //PORTB |= 1;
#define OVERRIDE_BUZZERPIN_OFF PORTC &= ~(1<<2); //PORTB &= ~1;

*****
```

Done Saving.



CAMERA GIMBAL

CONFIG.H

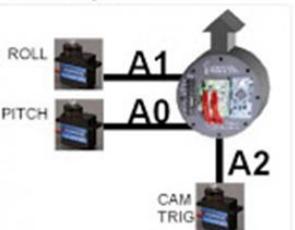


Photo for illustration purpose only.

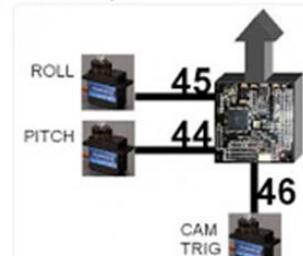
Pure stabilized gimbal system

For this setup, you need a GYRO + ACC setup (not possible with a gyro only setup).
A RC system is optional for this setup.

for Arduino 328p:



for arduino mega:



MultiWii - config.h | Arduino 1.8.2
File Edit Sketch Tools Help

```
/*
 * MultiWii - config.h | Arduino 1.8.2
 *
 * File Edit Sketch Tools Help
 */

MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp send

/*
 * if you want to preset min/middle/max values for servos right after flashing, because of limited physical
 * room for servo travel, then you must enable and set all three following options
 */
#define SERVO_MIN {1020, 1020, 1020, 1020, 1020, 1020, 1020}
#define SERVO_MAX {2000, 2000, 2000, 2000, 2000, 2000, 2000}
#define SERVO_MID {1500, 1500, 1500, 1500, 1500, 1500, 1500} // (*)
#define FORCE_SERVO_RATES {30,30,100,100,100,100,100} // 0 = normal, 1= reverse

/********************************************* cam_stabilisation *****************************************/
/* The following lines apply only for a pitch/roll tilt stabilization system. Uncomment the first or second line to activate it */
#define SERVO_MIX_TILT
//define SERVO_TILT

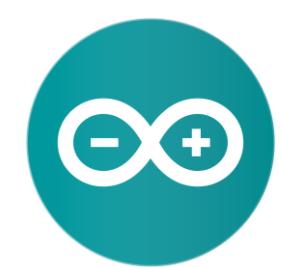
/* camera trigger function : activated via Rc Options in the GUI, servo output=A2 on promini */
// trigger interval can be changed via (*GUI*) or via AUX channel
#define CAMTRIG
#define CAM_TIME_HIGH 1000 // the duration of HIGH state servo expressed in ms

/********************************************* Airplane *****************************************/
//define USE_THROTTLESERVO // For use of standard 50Hz servo on throttle.

#define FLAPPERONS AUX4 // Mix Flaps with Ailerons.
#define FLAPPERON_EP { 1500, 1700 } // Endpoints for flaps on a 2 way switch else set {1020,2000} and program in radio.
#define FLAPPERON_INVERT { -1, 1 } // Change direction on flaperons { Wing1, Wing2 }

#define FLAPS // Traditional Flaps on SERVO3.
```

259 Generic ESP8266 Module, 80 MHz, Flash, ck, 26 MHz, 40MHz, OIO, 512K (no SPIFFS), 2, v2 Lower Memory, Disabled, None, Only Sketch, 115200 on COM24 6:16 PM 11/08/2020



CONFIG.H

Arm/DisArm

Option for combination stick command to start and stop the Vehicle

Note: Combination stick only works on some vehicles configuration . Others uses Aux Arm switch

Arm Only When Flat
Is a safety option not to arm when the drone is not level prevent starting up when on a slope or when moving

```
MultiWii - config.h | Arduino 1.8.2
File Edit Sketch Tools Help
MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp Sensors.h

/*
 * NEW: not used anymore for servo coptertypes <== NEEDS FIXING - MOVE TO WIKI */
#define YAW_DIRECTION 1
//#define YAW_DIRECTION -1 // if you want to reverse the yaw correction direction

#define ONLYARMWHENFLAT //prevent the copter from arming when the copter is tilted

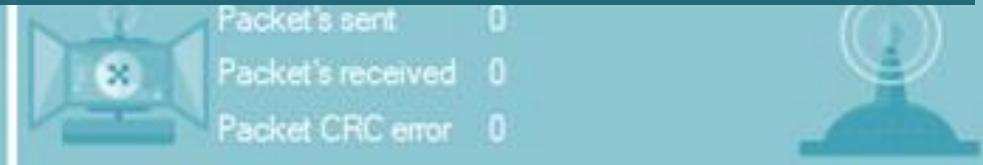
/********************* ARM/DISARM ********************/
/* optionally disable stick combinations to arm/disarm the motors.
 * In most cases one of the two options to arm/disarm via TX stick is sufficient */
#define ALLOW_ARM_DISARM_VIA_TX_YAW
//#define ALLOW_ARM_DISARM_VIA_TX_ROLL

/********************* SERVOS ********************/
/* info on which servos connect where and how to setup can be found here
 * http://www.multiwii.com/wiki/index.php?title=Config.h#Servos\_configuration
 */

/* Do not move servos if copter is unarmed
 * It is a quick hack to overcome feedback tail wiggles when copter has a flexible
 * landing gear
 */
#define DISABLE_SERVOS_WHEN_UNARMED

/* if you want to preset min/middle/max values for servos right after flashing, because of limited physical
 * room for servo travel, then you must enable and set all three following options */
#define SERVO_MIN {1020, 1020, 1020, 1020, 1020, 1020, 1020, 1020}
#define SERVO_MAX {2000, 2000, 2000, 2000, 2000, 2000, 2000, 2000}
#define SERVO_MTD {1500, 1500, 1500, 1500, 1500, 1500, 1500, 1500} // (*)
```

Works for Vehicle Example : Rudder Stick Left to Rudder Arm Stick Right to Disarm
Note: motors will spool up to Idle Speed





FLIGHT DECK – IF THIS DOESN'T LOOK RIGHT CHECK YOUR SENSORS ORIENTATION AGAIN USING THE SENSOR GRAPH

TELEMETRY CONNECTION SEE YOUR CHECK YOUR BLUETOOTH RADIO OR USB ON WHERE IS THE VIRTUAL COM PORT IS

COMPASS (MAG AND GYRO)

PWM OUTPUT INDICATOR

PWM INPUT INDICATOR

SAVE CONFIG

FLIGHT & GPS LOGS

ALTITUDE (BARO)

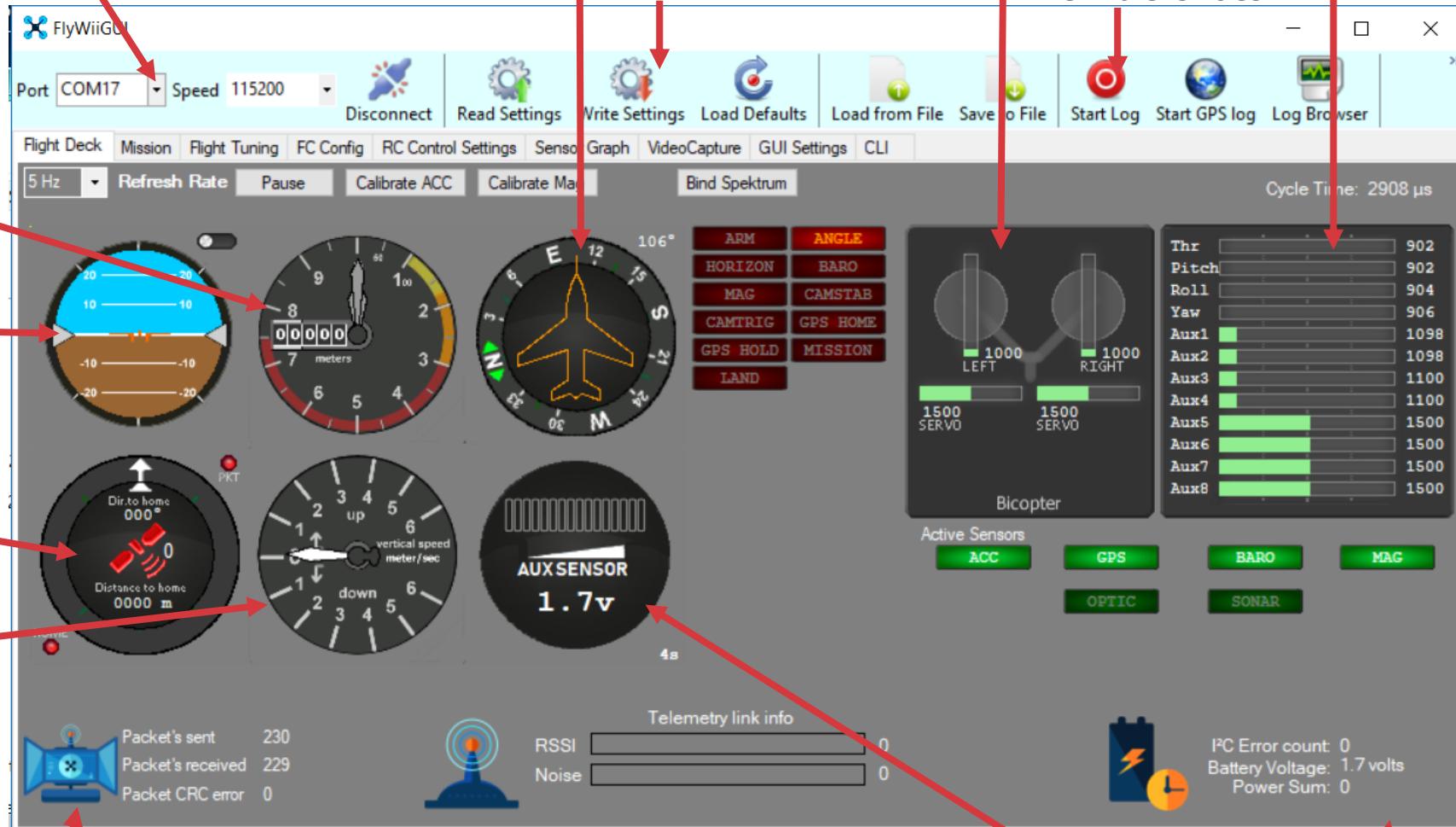
ATTITUDE (ARTIFICIAL HORIZON)
(GYRO XYZ AND ACC XYZ)

GPS SATELLITE COUNT
(4 SATS FOR 3D FIX – IDEAL 7 SATS)

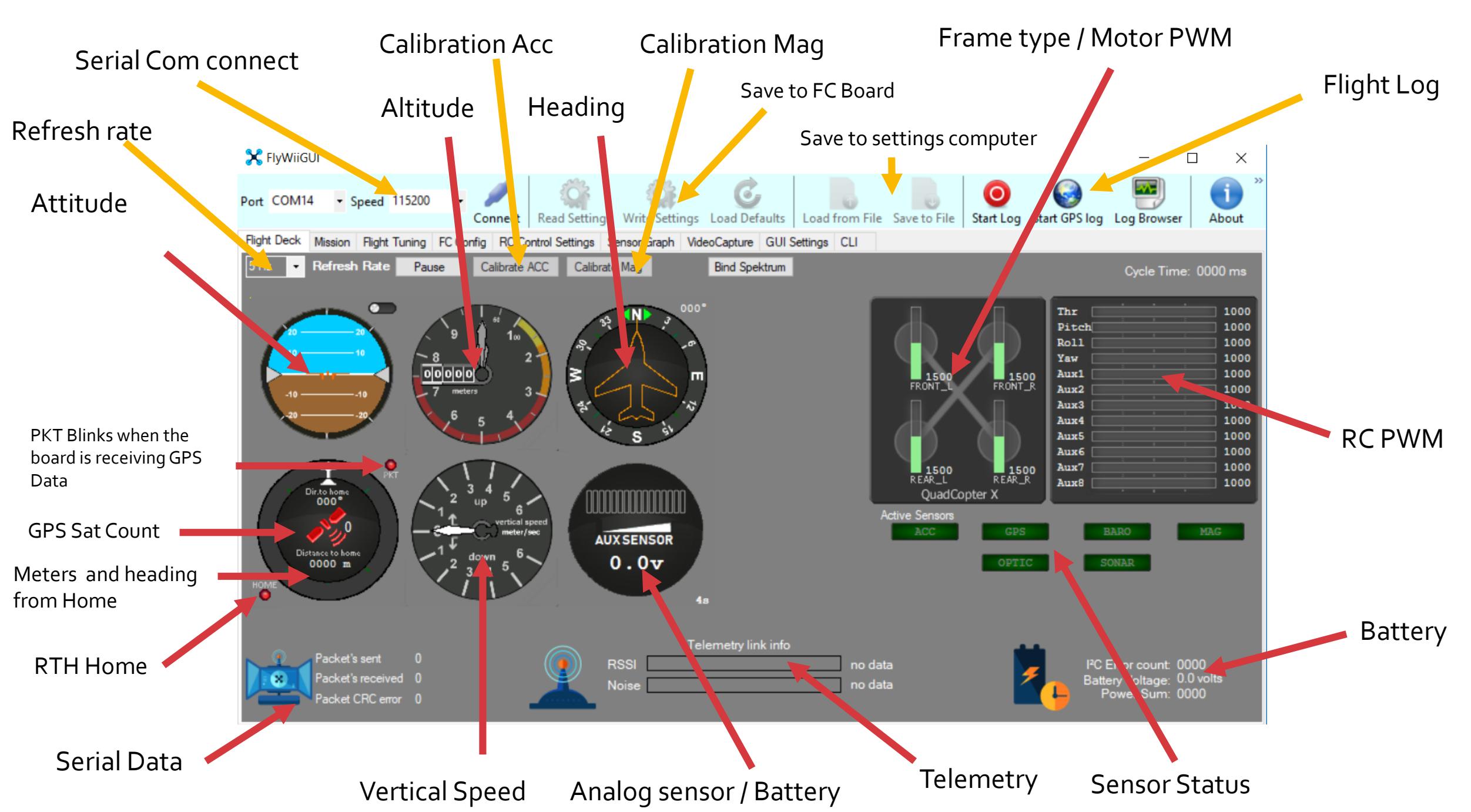
VERTICAL SPEED INDICATOR
(ACC Z AXIS)

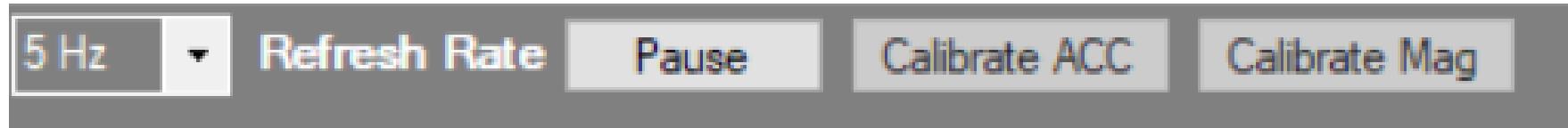
PACKETS STATUS

(IF THE ERROR NUMBERS ARE HIGH PLS
CHECK YOUR TELEMETRY CONFIG)



POWER STATUS / AUX SENSOR
ALSO KNOWN AS FUEL GAUGE
(VBAT)



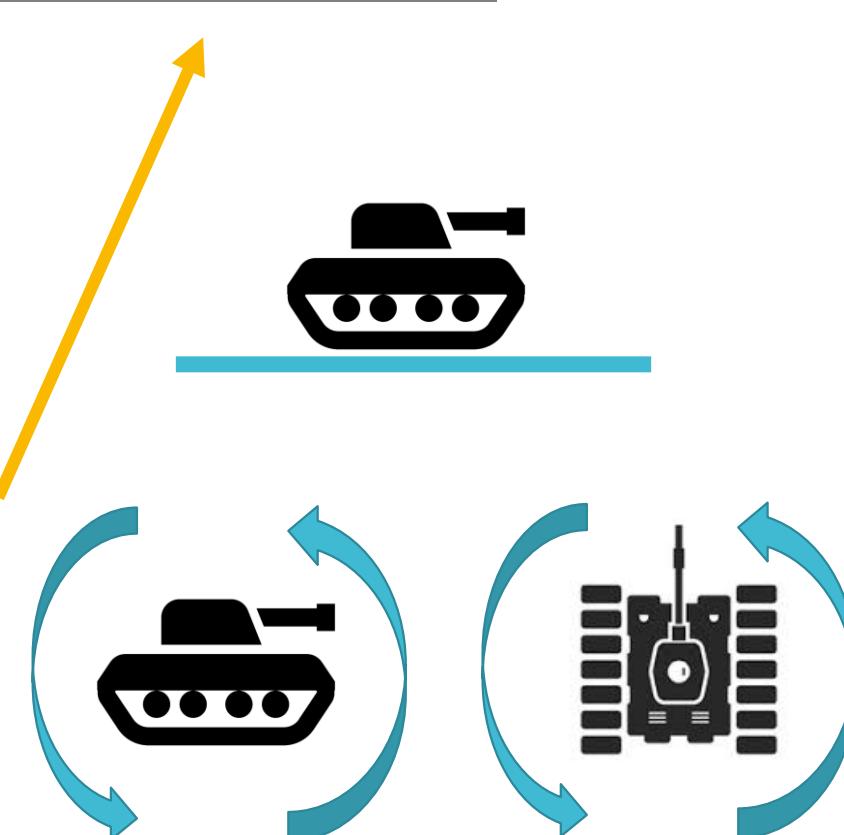


Refresh Rate . Telemetry update speed

Acc Calibration . Set the vehicle down on a level surface . Away from any metal objects for 10 secs.

Mag Calibration . rotate the vehicle 360 degrees in all axis within 30sec - 1 min. while the blue Led flashes

Mag Calibration must be perform when running your vehicle in a new location for the first time. Pls verified the Compass if the drone heading matches your compass app in your phone.

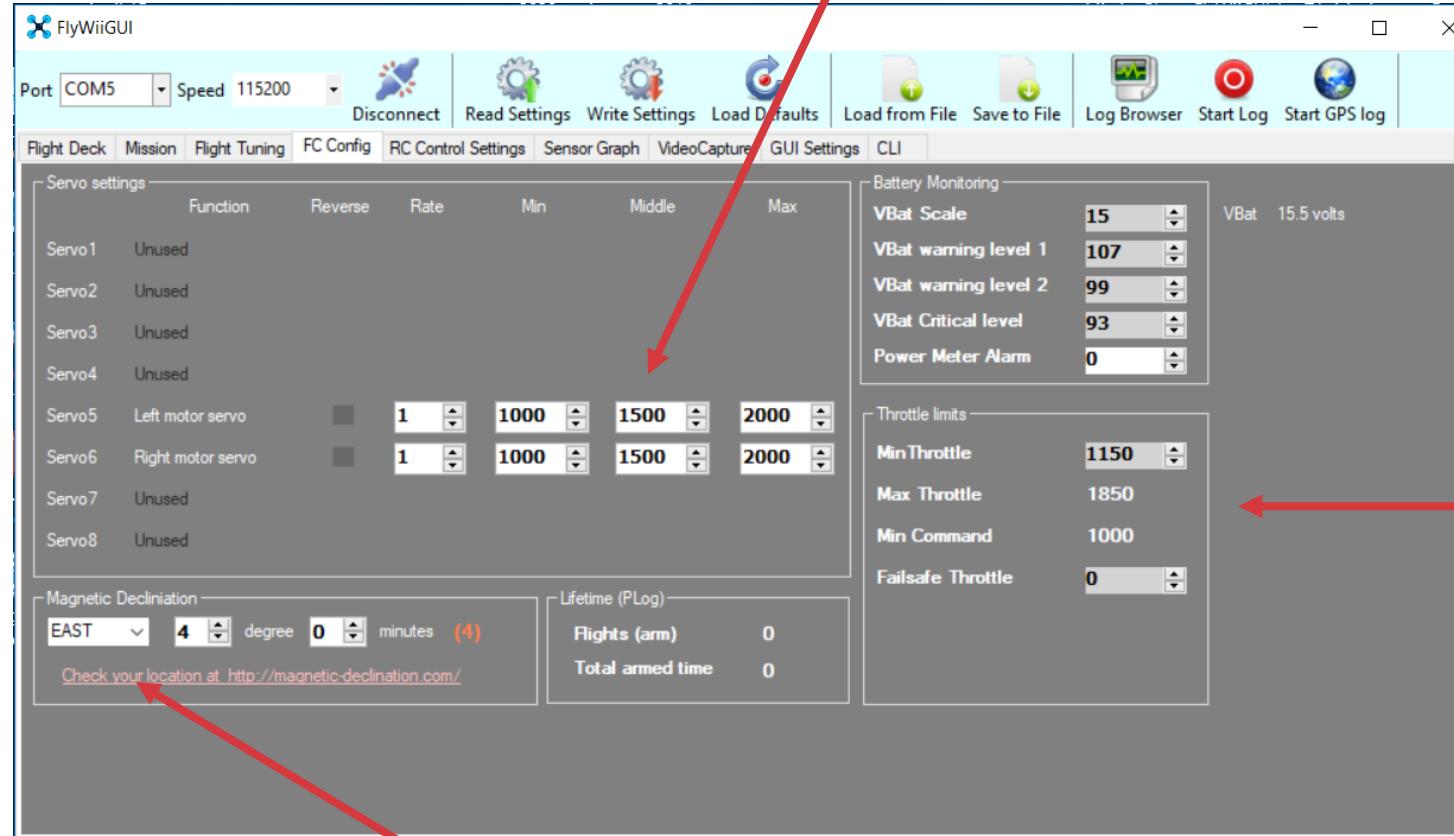


These Calibration must be perform after Parameter updates after Flushing the firmware
Blue LED would flash during these calibration processes

Other Navigation Functions

FC Config

SERVO REVERSE OPTION – IF THE SERVO / REVERSABLE ESCS
IS OPERATING IN THE WRONG DIRECTION FROM CONTROLS



MOTOR THROTTLE RANGE PWM TO THE MOTOR
THIS ALSO CONTROLS THE MOTOR IDLE SPEED ON ARM

IMPORTANT TO KNOW THE MAGNETIC DECLINATION OF YOUR REGION

THIS AID ANY AUTONOMOUS FUNCTION THAT REQUIRES COMPASS

- HEADING HOLD
- GPS HOLD
- RTH
- MISSION

CALIBRATE COMPASS AT THE FLIGHT DECK TAB AFTER SETTING THIS UP



FLYWII GUI

Battery Monitoring

VBat Scale	120
VBat warning level 1	110
VBat warning level 2	110
VBat Critical level	109
Power Meter Alarm	0

VBat 15.4 volts

Battery Cell Count

4s	▼
1s	
2s	
3s	
4s	
5s	
6s	
7s	
8s	
9s	
10s	

(FC CONFIG TAB)

BATTERY MONITORING

VBAT SCALE - ADJUST THIS TO MATCH THE BATTERY VOLTAGE OUTPUT USING THE VOLTAGE ALARM INDICATOR

VBAT WARNING LEVEL – IDENTIFY THE NOTICE WHEN THE BATTERY DROPS TO THIS VOLTAGE

(GUI SETTINGS TAB)

BATTERY CELL COUNT- ADJUST THIS DEPENDING ON THE NUMBER OF CELLS

THIS BOARD SUPPORTS 2S-4S BATTERY

BATTERY

Vehicle Tuning

FlyWiiGUI

Port COM17 Speed 115200

Disconnect Read Settings Write Settings Load Defaults Load from File Save to File Start Log Start GPS log Log Browser

Flight Deck Mission Flight Tuning FC Config RC Control Settings Sensor Graph VideoCapture GUI Settings CLI

Roll
P 3.0 I 0.030 D 23
Pitch P 3.0 I 0.030 D 23
Yaw P 6.8 I 0.045 D 0
Altitude P 6.6 I 0.015 D 14
PosHold P 0.15 I 0.00
PosHoldRate P 4.4 I 0.14 D 0.053
Navigation Rate P 3.5 I 0.33 D 0.083
Level P 14.0 I 0.010 D 100
Mag P 18.0

RC Expo 0.00 Rate: 0.90 Expo: 0.00

RC Rate 0.90

Thr. MID 0.50 Mid: 0.50 Expo: 0.00

Thr. EXPO 0.00

Rates/Expo
Roll/Pitch RATE 0.00
Yaw RATE 0.00
Throttle PID attenuation 0.50

Navigation settings -

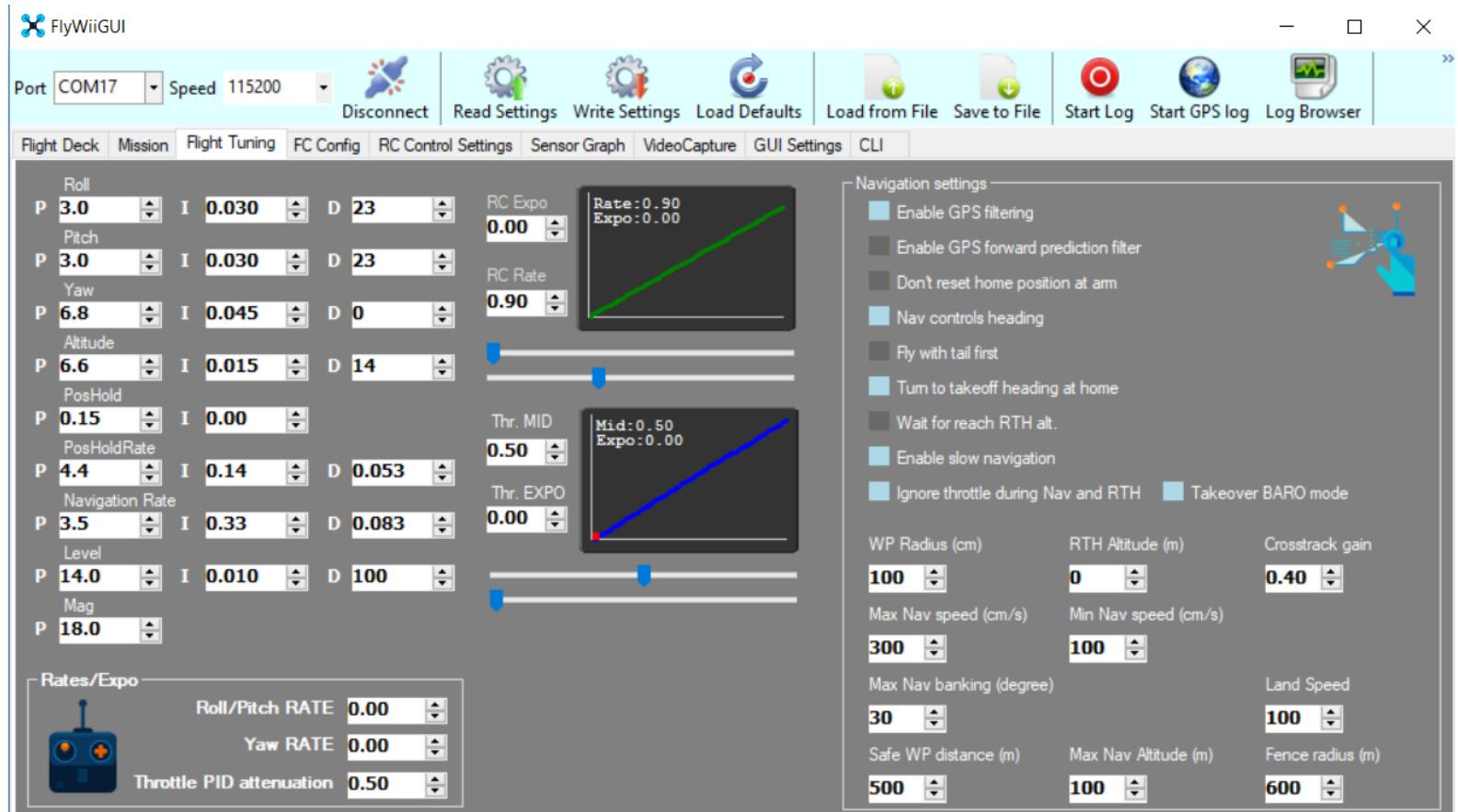
- Enable GPS filtering
- Enable GPS forward prediction filter
- Don't reset home position at arm
- Nav controls heading
- Fly with tail first
- Turn to takeoff heading at home
- Wait for reach RTH alt.
- Enable slow navigation
- Ignore throttle during Nav and RTH
- Takeover BARO mode

WP Radius (cm) 100 RTH Altitude (m) 0 Crosstrack gain 0.40

Max Nav speed (cm/s) 300 Min Nav speed (cm/s) 100

Max Nav banking (degree) 30 Land Speed 100

Safe WP distance (m) 500 Max Nav Altitude (m) 100 Fence radius (m) 600



Stability	Roll,Pitch,Yaw Gyro	PID : Level of your Gyro
Horizon / Level Mode	X,Y Accelerometer	PID : X ,Y Axis Level of your Accelerometer
Heading Lock	Compass/Mag	PID : heading of your magnetometer Calibration
Altitude Hold	Barometer / Z	PID : Barometer and Z accelerometer / not applicable for surface vehicles
Position Hold	GPS Pos	PID : sensitivity of GPS position reaction
Navigation Rate	GPS Nav	

Understanding impact of P, I and D

P : this is the amount of corrective force applied to return the **Vehicle** back to its initial position

The amount of force is proportional to a combination of the deviation from initial position minus any command to change direction from the controller input. A higher P value will create a stronger force to resist any attempts to change it's position. If the P value is too high, on the return to initial position, it will overshoot and then opposite force is needed to compensate. This creates an oscillating effect until stability is eventually reached or in severe cases becomes completely destabilized.

I : this is the time period for which the angular change is sampled and averaged

The amount of force applied to return to initial position is increased by the I factor the longer the deviation exists until a maximum force value is reached. A higher I will increase the angular hold capability.

D : this is the speed at which the **Vehicle** is returned to its original position

Increasing value for D: Improves the speed at which deviations are recovered

With fast recovery speed comes a higher probability of overshooting and oscillations
Will also increase the effect of P

P – proportional

P provides a proportional amount of corrective force based upon the angle of error from desired position. The larger the deviation, the larger the corrective force.

A higher P value will create a stronger force to return to desired position. If the P value is too high, on the return to initial position, it will overshoot and then opposite force is needed to compensate. This creates an oscillating effect until stability is eventually reached or in severe cases, the overshoot becomes amplified and the multi-rotor becomes completely destabilized.

Increasing value for P :It will become more solid/stable until P is too high where it starts to oscillate and lose control. You will notice a very strong resistive force to any attempts to move the Multi-Rotor

Decreasing value for P: It will start to drift in control until P is too low when it becomes very unstable. Will be less resistive to any attempts to change orientation

I – Integral

“I” gain provides a variable amount of corrective force based upon the angle of error from desired position.

The larger the deviation and / or the longer the deviation exists, the larger the corrective force. It is limited to prevent becoming excessively high.

A higher I will increase the heading hold capability

Increasing value for I: Increase the ability to hold overall position, reduce drift due to unbalanced frames etc

Decreasing value for I: Will improve reaction to changes, but increase drift and reduce ability to hold position

D- Divide / Derivative

This moderates the speed at which the Vehicle is returned to its original position.

A lower D will mean the Vehicle will snap back to its initial position very quickly

Increasing value for D: Dampens changes. Slower to react to fast changes

Decreasing value for D: Less dampening to changes. Reacts faster to changes

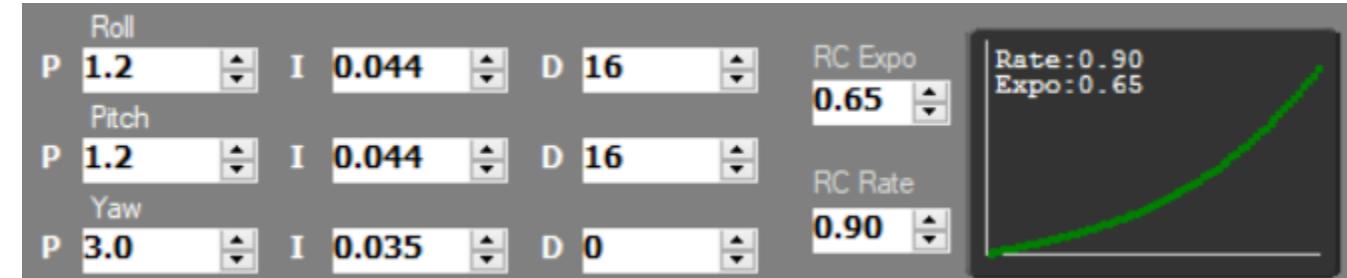
Aerobatic flight: Lower D

Gentle smooth flight: Increase D



Basic PID Tuning – on the ground

- 1 - Set PID to the designers default recommended settings.
- 2 - Hold the vehicle securely and safely in the air.
- 3 - Increase throttle to the hover point where it starts to feel light.
- 4 - Try to lean the Vehicle down onto each motor axis. You should feel a reaction against your pressure for each axis.
- 5 - Change P until it is difficult to move against the reaction. Without stabilization you will feel it allow you to move over a period of time. That is OK
- 6 - Now try rocking the vehicle. Increase P until it starts to oscillate and then reduce a touch.
- 7 - Repeat for Yaw Axis. Your settings should now be suitable for flight tuning.



Click on Write settings
after changes made in
any of the parameters to
save

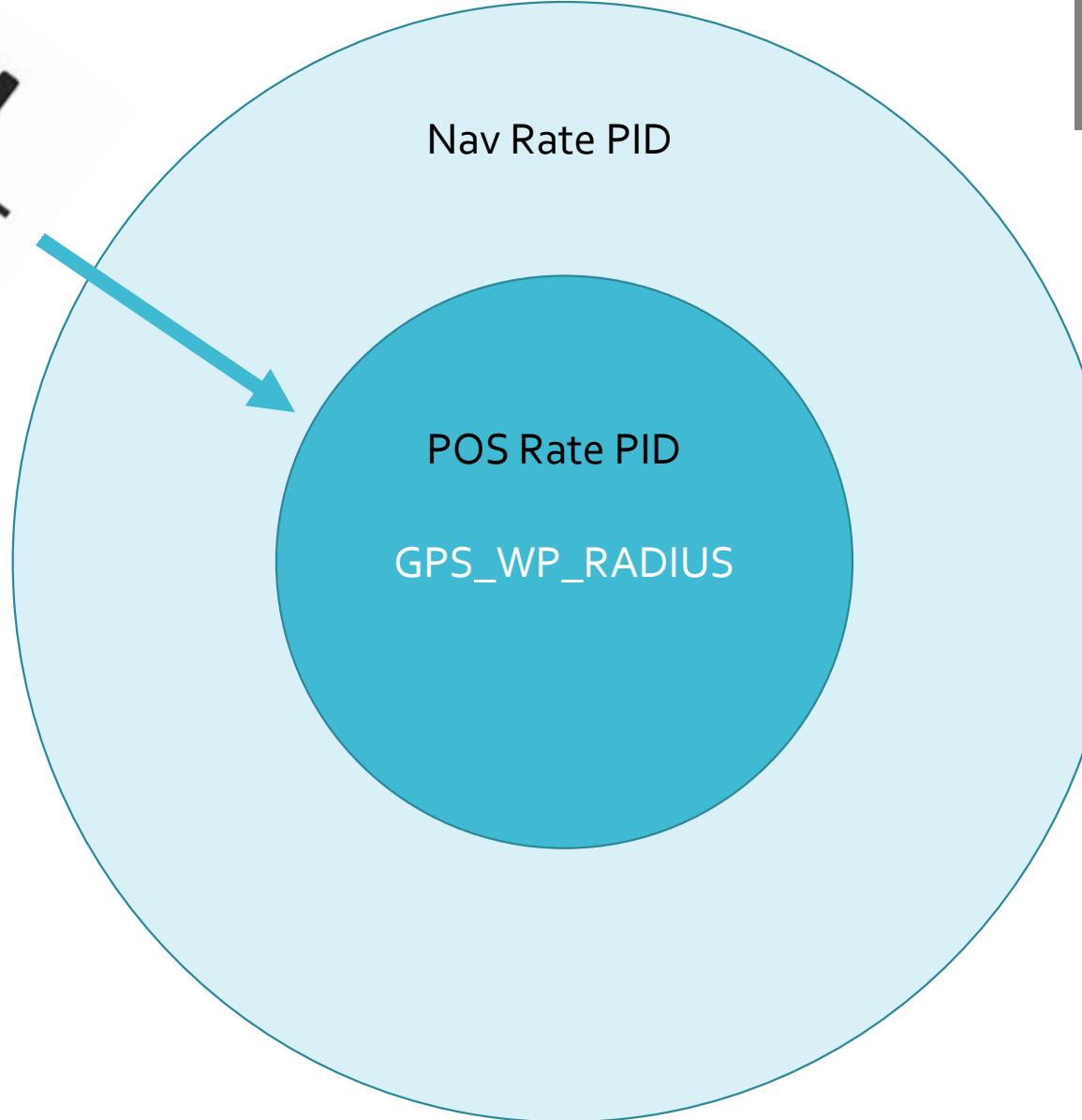
For Mega 2560 + GPS Pos Rate PID controller & Pos Rate PID Tuning

- Pos Rate PID controller
- Pos Rate PID Tuning
- The Pos Rate PID controller takes the commanded speed output from the Pos PI controller and commands an attitude in order to maintain the position hold location. This PID controller should be tuned before adjusting the Pos PI controller.
- The Pos Rate PID settings control how the attitude of the Vehicle is changed in order to move towards the desired hold location.
- The speed of movement is controlled by the Pos PI controller, while the attitude of the Vehicle is controlled by Pos Rate.
- When the Vehicle is within the defined distance of the hold location or waypoint (set by GPS_WP_RADIUS in config.h) the Pos Rate PID is used, when further away from the location the Nav Rate PID is used to return to within the defined waypoint radius.

For Mega 2560 + GPS Pos Rate PID controller & Pos Rate PID Tuning

- The Pos Rate PID controller takes the commanded speed output from the Pos PI controller and commands an attitude in order to maintain the position hold location. This PID controller should be tuned before adjusting the Pos PI controller.
- The Pos Rate PID settings control how the attitude of the Vehicle is changed in order to move towards the desired hold location.
- The speed of movement is controlled by the Pos PI controller, while the attitude of the multirotor is controlled by Pos Rate.
- When the Vehicle is within the defined distance of the hold location or waypoint (set by `GPS_WP_RADIUS` in `config.h`) the Pos Rate PID is used, when further away from the location the Nav Rate PID is used to return to within the defined waypoint radius.

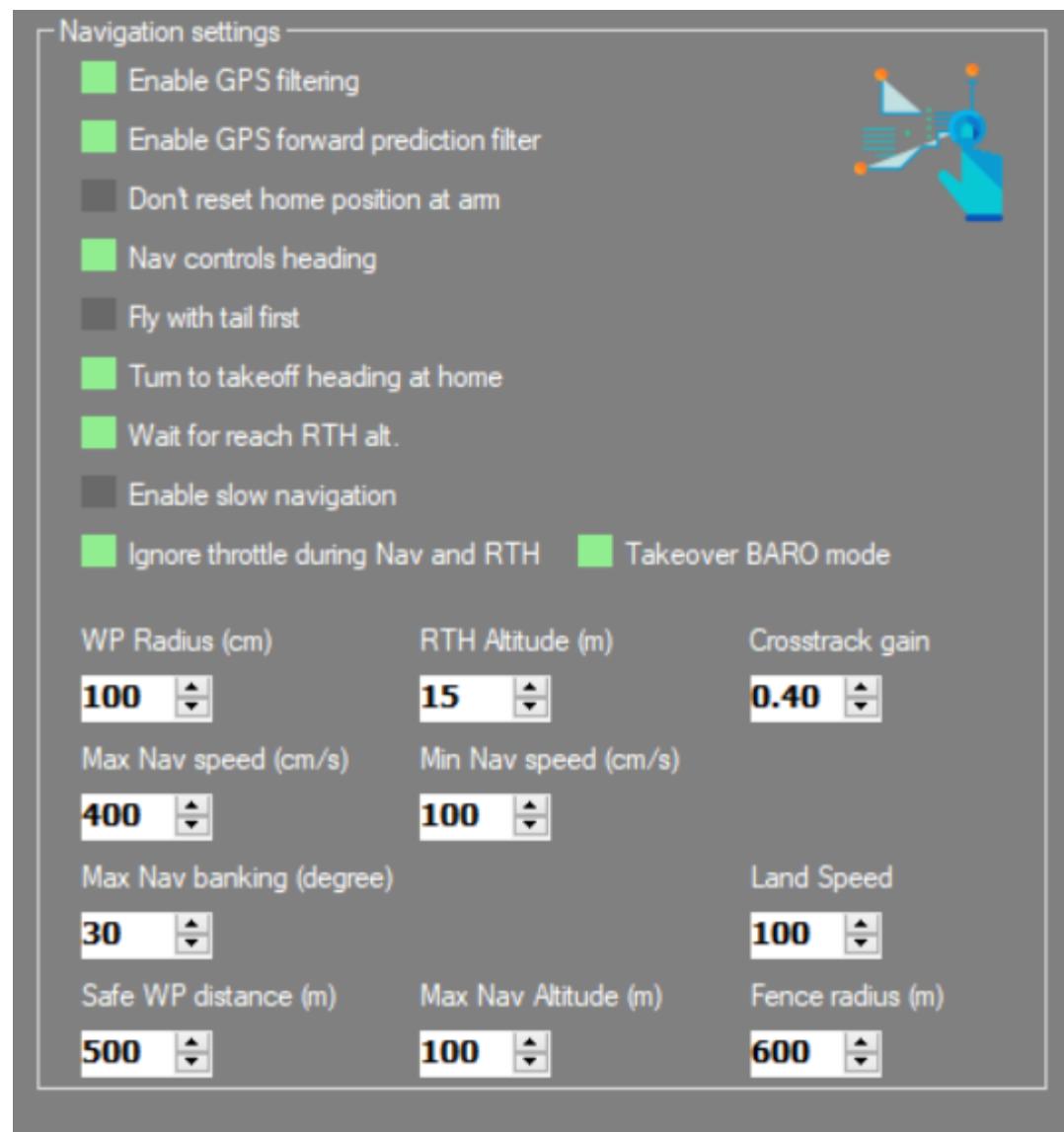
- To tune the Pos Rate PID, initially set P, I and D values to 0.
- Gradually increase P until the Vehicle begins to position hold with some drift.
- Gradually increase D until the Vehicle responds more quickly to undesired changes in attitude caused by the wind. If this value is set too high you will see oscillations or sudden jerking in pitch and roll motion.
- If needed, gradually increase I value to allow the PID controller to compensate for long lasting error, ie if it is being blown by the wind.



PosHold	P 0.15	I 0.00
PosHoldRate	P 3.4	I 0.14 D 0.053
Navigation Rate	P 2.5	I 0.33 D 0.083

The Navigation responds is the vehicle will orientate to the direction of the waypoint It will proceed to drive straight until it reach the WP_Radius and stop With multiple waypoint it will repeat the process till it completes the mission

Other Navigation Functions



WP Radius – the radius of the area the Pos PID will trigger it has reached the waypoint

Max Nav Speed – Maximum speed the Vehicle travel between waypoints (too fast and you likely over shoot your target) **for first mission flight test Nav speed of 100cm/s with ("Enable Slow Navigation "Active)**

Min Nav Speed – the speed the drone travel when within the WP Radius

RTH Altitude – (Zero Value for surface vehicle)

Max Nav Banking – the max allowable pitch and roll the drone will be set to while traveling between waypoints (tune this along with Max Nav Speed to take account with Environment conditions) **(not applicable for surface vehicle)**

Max Nav Altitude – Max altitude the drone is capped to fly at (Zero Value for surface vehicle)

Land Speed – speed of descending for Landing cm/s **(not applicable for surface vehicle)**

Safe WP Distance – max distance between waypoint before its null out

Fence Radius – Geo Fence to keep the drone within the perimeter in relation to home position

CrossTrack gain - this tunes the GPS and Nav sensitivity

GPS Filtering – used to enhance GPS accuracy

GPS Forward Prediction Filter – predicting the Vehicle's location and to compensate for lag . (optional) – not necessary for most applications

Navigation settings –

- Enable GPS filtering
- Enable GPS forward prediction filter
- Don't reset home position at arm
- Nav controls heading
- Fly with tail first
- Turn to takeoff heading at home
- Wait for reach RTH alt.
- Enable slow navigation
- Ignore throttle during Nav and RTH Takeover BARO mode

WP Radius (cm)	RTH Altitude (m)	Crosstrack gain
100	15	0.40
Max Nav speed (cm/s)	Min Nav speed (cm/s)	
400	100	
Max Nav banking (degree)	Land Speed	
30	100	
Safe WP distance (m)	Max Nav Altitude (m)	Fence radius (m)
500	100	600



Don't Reset Home position at Arm – this retains the home position where you first plug power on your Vehicle

Nav Controls Heading – this points the Vehicle to its next waypoints (Keep this on) exception for omnidirectional vehicles

Fly tail first – makes the Vehicle drive reverse

Turn take off heading at Home – when drone arrives at home position it orientates to its heading right after arming

Wait to reach RTH - this works with RTH altitude command which the drone would climb to the said altitude before initiating the flight to home position (turn off for surface vehicle) **(not applicable for surface vehicle)**

Enable slow navigation – this works with keeping the drone to its **Min Nav speed**

Ignore throttle and Take over Baro – Not applicable



WiiGUI

- □ ×[Flight Deck](#) [Mission](#) [Flight Tuning](#) [FC Config](#) [RC Control Settings](#) [Sensor Graph](#) [VideoCapture](#) [GUI Settings](#) [CLI](#)

	AUX1	AUX2	AUX3	AUX4
ARM	L M H	L M H	L M H	L M H
ANGLE	[]	[]	[]	[]
HORIZON	[]	[]	[]	[]
BARO	[]	[]	[]	[]
MAG	[]	[]	[]	[]
HEADFREE	[]	[]	[]	[]
HEADADJ	[]	[]	[]	[]
CAMSTAB	[]	[]	[]	[]
CAMTRIG	[]	[]	[]	[]
GPS HOME	[]	[]	[]	[]
GPS HOLD	[]	[]	[]	[]
MISSION	[]	[]	[]	[]
LAND	[]	[]	[]	[]

RC Control Settings

Use Aux switch to setup flight modes and Navigation functions

ARM – this is option should you decided to use a Aux switch oppose to the Combination Stick input to Arm/Disarm Vehicle

BARO – Not applicable

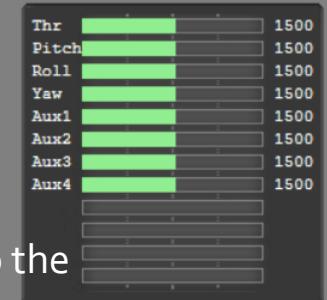
MAG – Heading Hold

HEADFREE – Course Lock regardless of orientation

GPS Home – Return to Home Vehicle returns to where its armed

GPS Hold – Hold Position

MISSION – run a waypoint mission



Live RC data

- Preferably to reserve 1 Aux channel as Arm switch as combination stick could make the vehicle move prematurely
- Then the alternative Aux for the Mission or RTH mode



Missions

Note: Only functional for Mega 2560 Boards with GPS

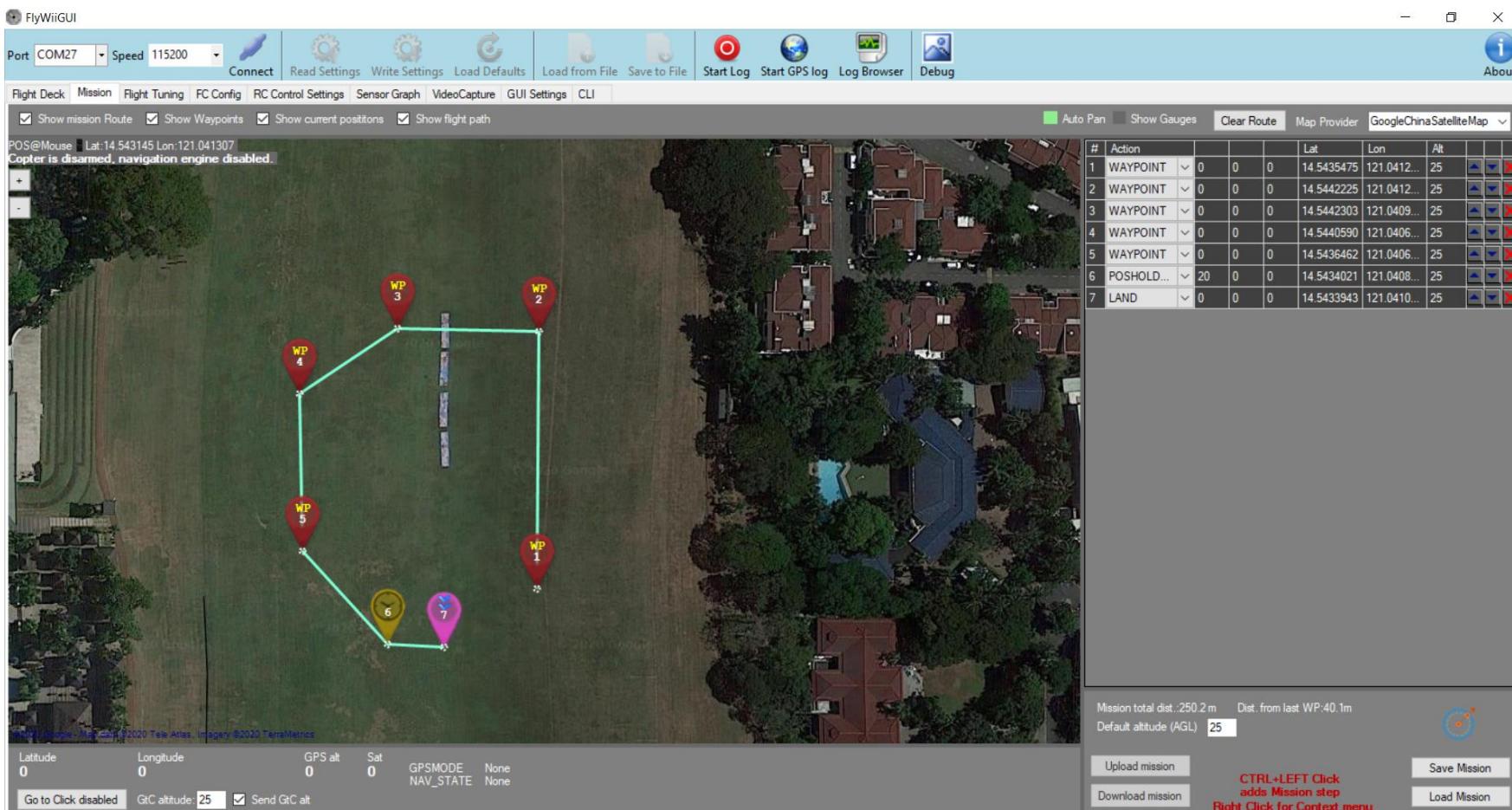
Waypoint – the Vehicle with travel between those points

Time PosHold – Vehicle will wait X number of 00:00:00 then move to the next waypoint

Unlimited PosHold – once the Vehicle reach this point it will hover and wait till you switch out of Mission mode

Land – the Vehicle Stop when has reach this point (Must be place at the end of the mission)

RTH – the Vehicle will go back to home position (Must be place at the end of the mission)



RC Control Setting Tab – activate Baro , Mag , Mission

To start mission takeoff aircraft in stabilize mode up to 1-2meter altitude then switch the aux switch to mission mode .

Any time you can switch out of it on hold or stabilize mode



Missions

Note: Only functional for Mega 2560 Boards with GPS

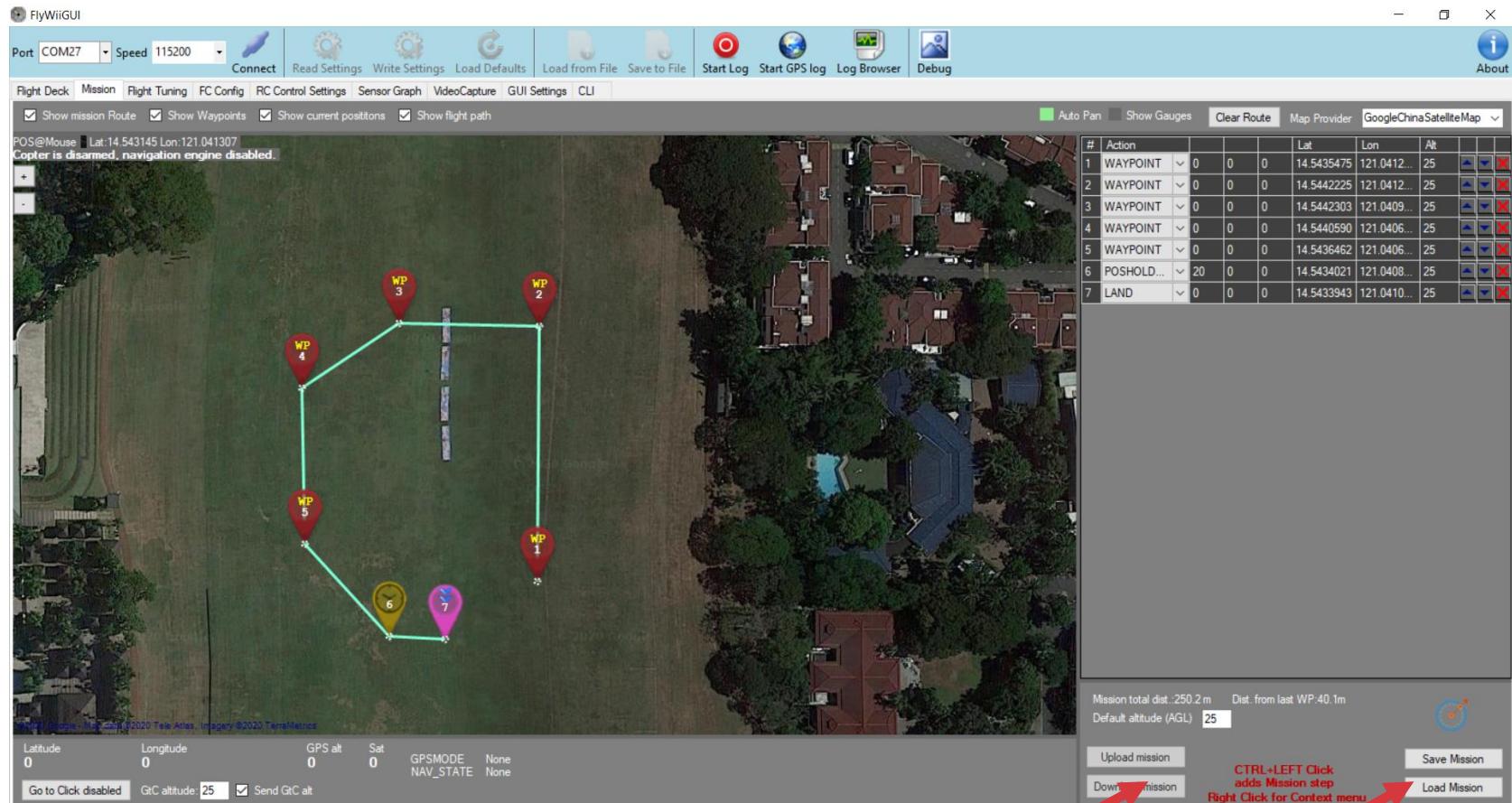
Waypoint – the Vehicle with travel between those points

Time PosHold – Vehicle will wait X number of 00:00:00 then move to the next waypoint

Unlimited PosHold – once the Vehicle reach this point it will hover and wait till you switch out of Mission mode

Land – the Vehicle Stop when has reach this point (**Must be place at the end of the mission**)

RTH – the Vehicle will go back to home position (**Must be place at the end of the mission**)



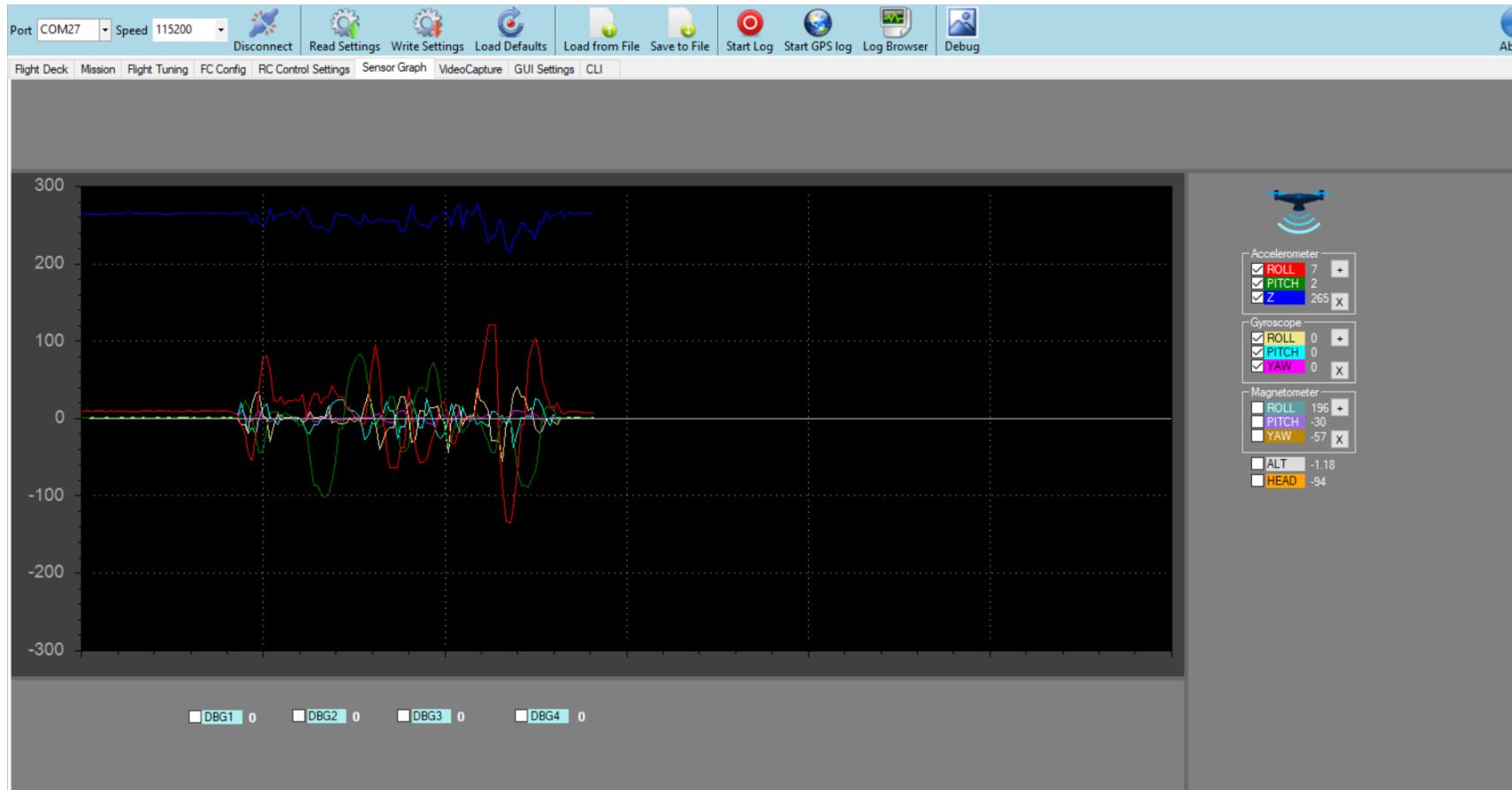
Mission upload to /download from Vehicle

Mission Save to /Open from File



Graphs and Sensors

Upload the sketch to the Arduino attach to the drone shield and open the FlywiiGUI sensor Graphs tab and hit connect to the appropriate COM your drone is connected to



the correct orientation

Roll Right + no#

Pitch nose down + No#

Z up + No#

Roll Right + no#

Pitch nose down + No#

Yaw Right +No#

Mag & HEAD degrees
corresponds to the compass

(0 degrees = North)

Alt up +no#

Example : if roll the drone to the right the Accelerometer and Gyroscope graphs would show positive numbers and to the Left Negative numbers

If Lift the drone up Vertically the accelerometer Z axis should shows positive numbers and altitude should show a climb in meters

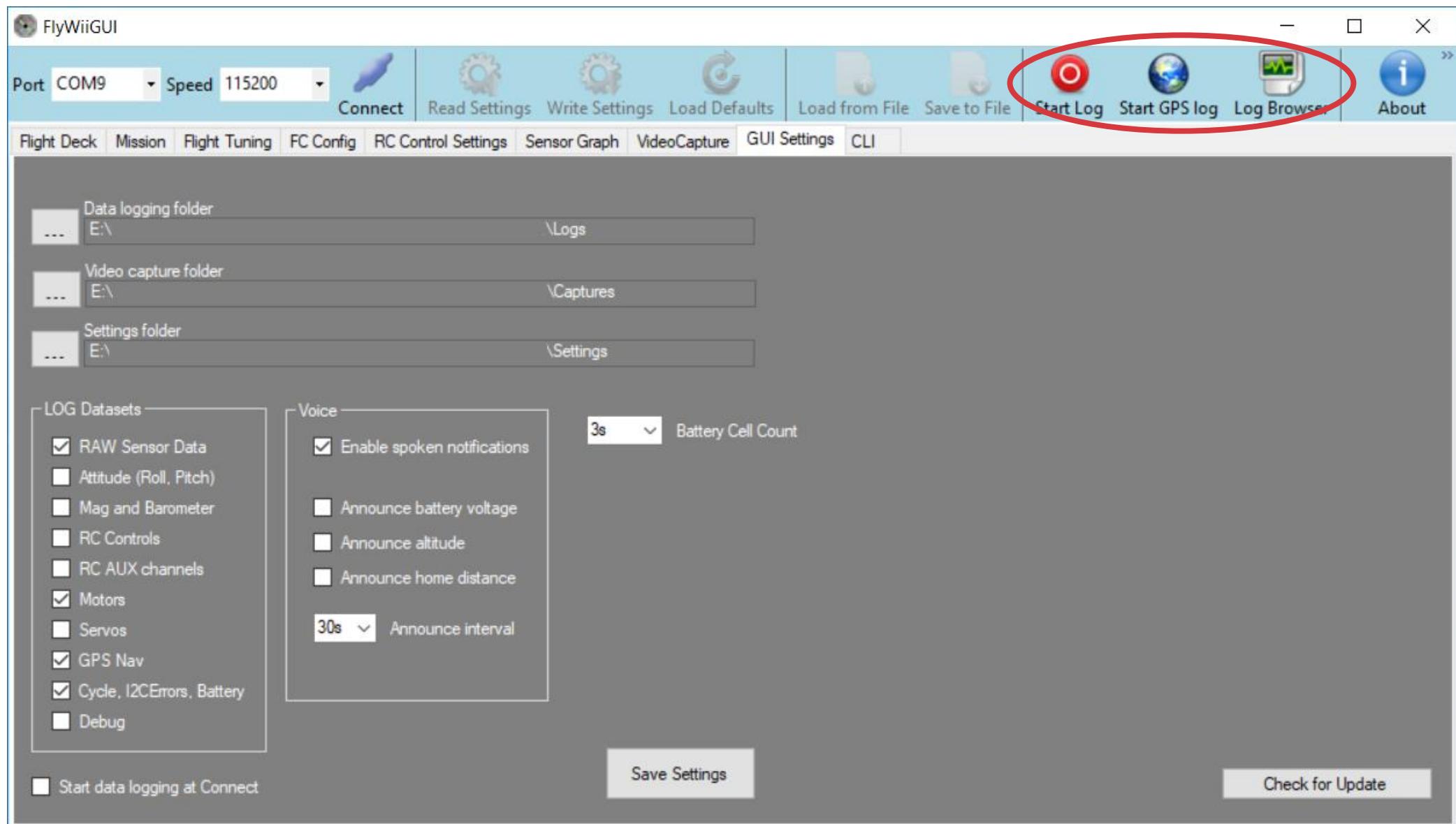


Graphs and Data Logging

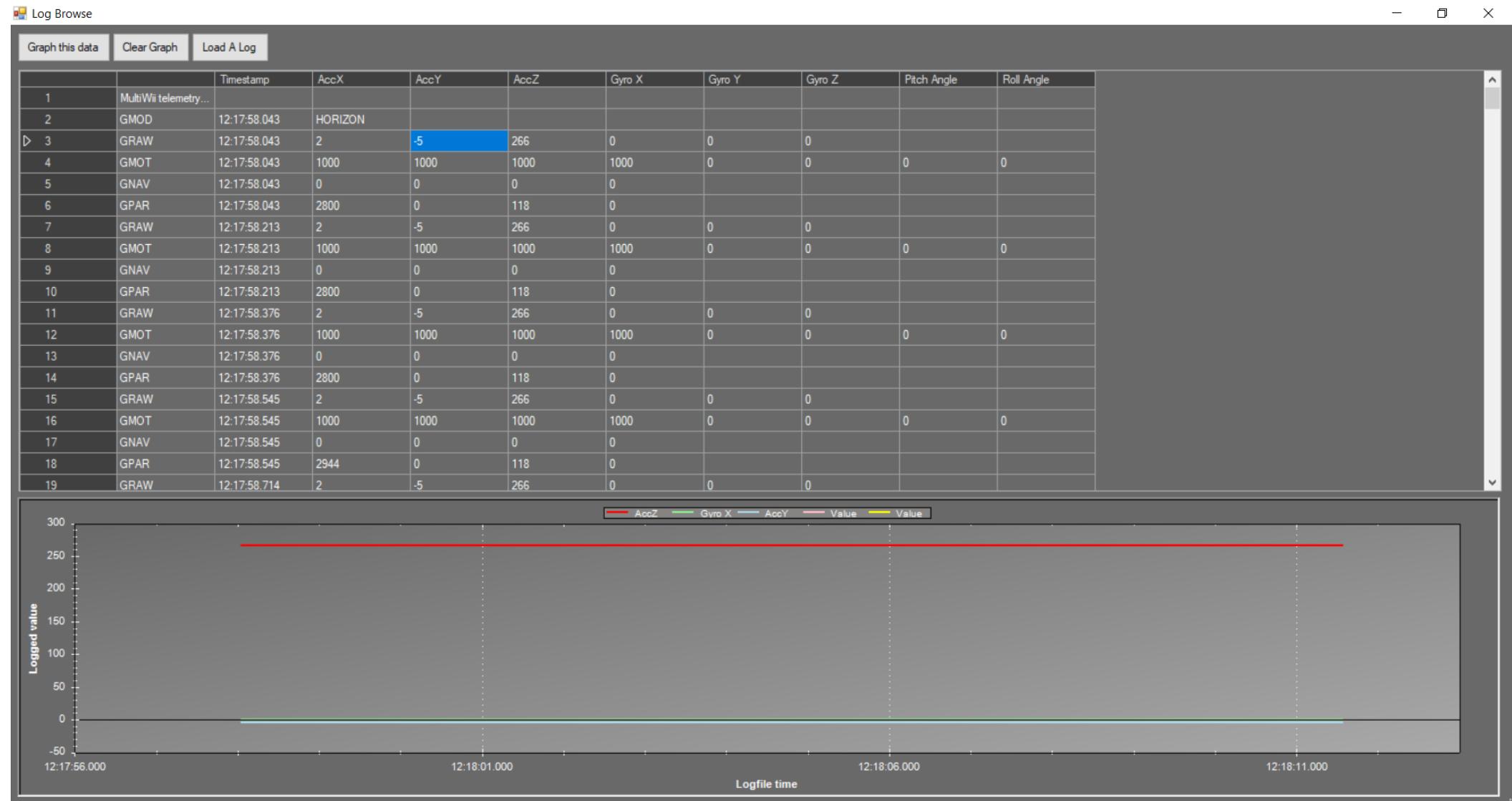
Telemetry Log

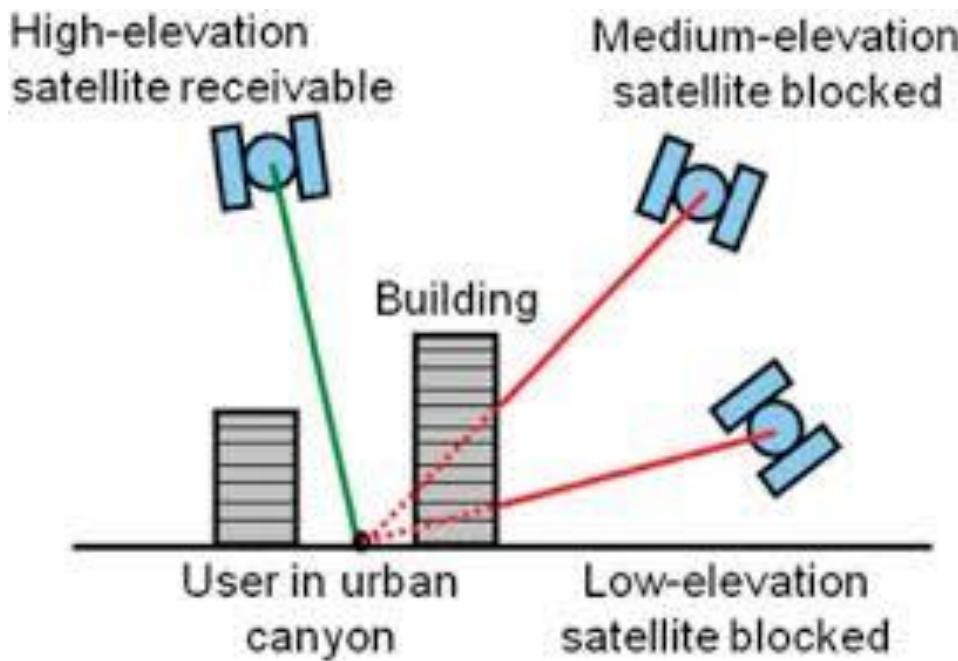


GUI Settings (where you save your PID ,Flight Logs and Video Logs)



Log Browser





Note : GPS require a clear open area to get a proper fix and accuracy minimum 7 satellites but 10+ are Ideal

operating next to a building can distort satellite signal deteriorating accuracy

Which in this case its better to not use GPS modes and operate Manual

And your much Done on your setup

Cannot Arm Motors

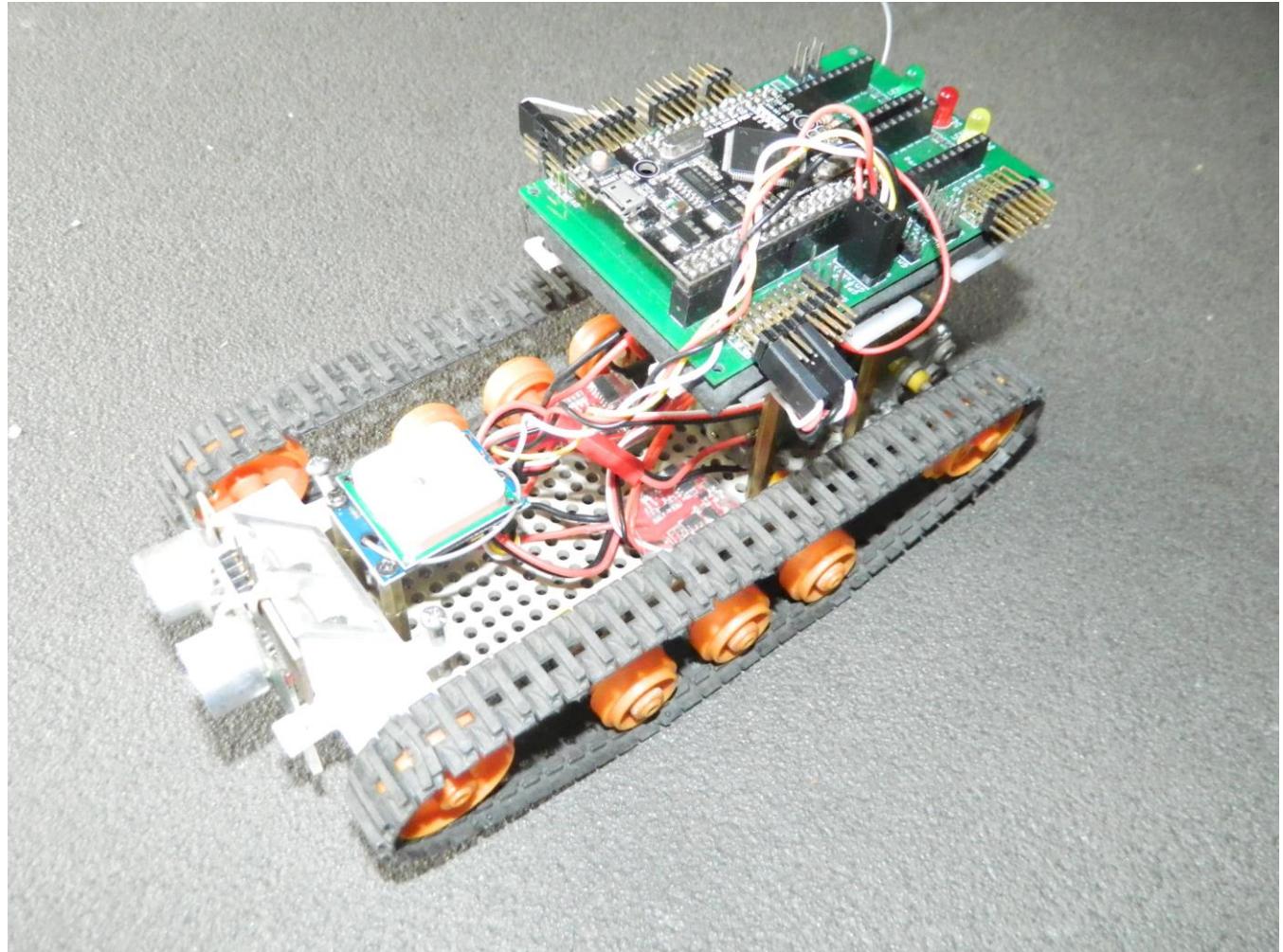
when on GPS Home , GPS Hold , Mission Flight modes & when USB is plugged in . (pls use Bluetooth telemetry)

You can Test with the Vehicle's wheels off the ground first

Pls calibrate ACC and Mag in the FlyWii GUI Dashboard

Ensure the compass is facing the correct orientation

Set Aux1 as Arm Switch at RC control



For surface vehicles at RC control settings

Set Aux1 as Arm Switch at RC control

Boats must be plug in on level surface first before placing on water

