

SYNERDUINO
KWAD SHIELD

V1.1

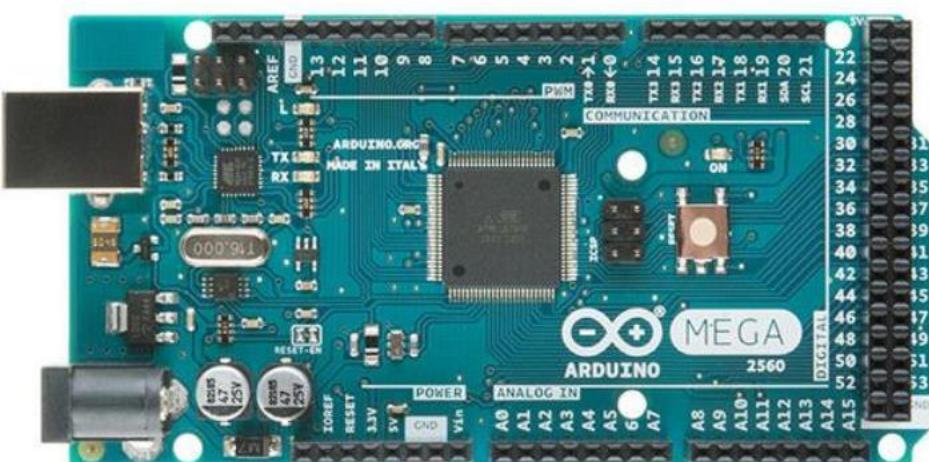
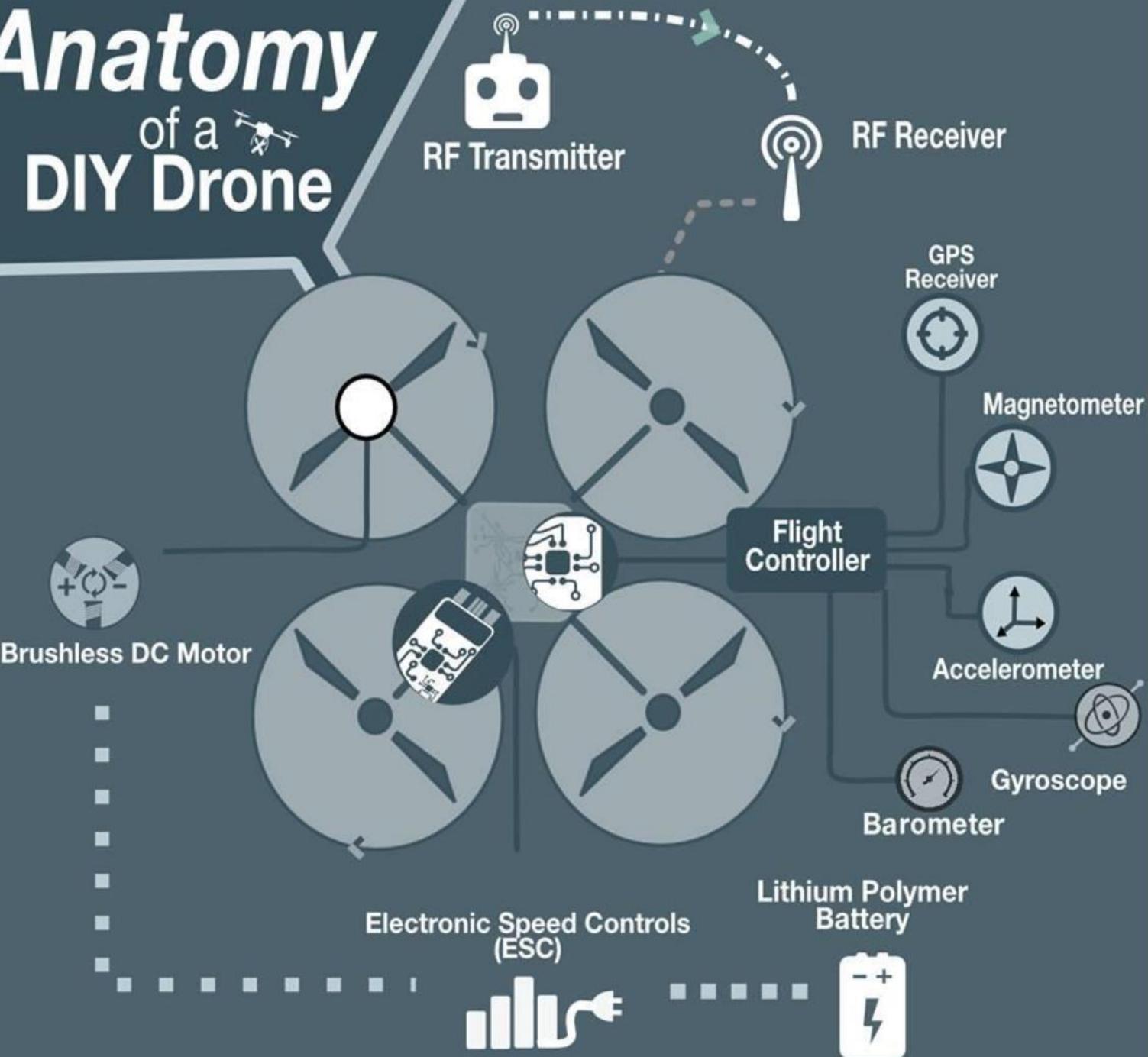




INTRODUCTION TO ARDUINO MULTIROTOR DRONES

- PREPARATION OF MATERIALS AND TOOLS NEEDED
- INSTALLATION NECESSARY APPLICATION FOR CONFIGURING THE MICRO CONTROLLER
- HOW DRONES OPERATE AND FLY
- PARTS AND COMPONENTS
- TYPES OF MULTICOPTER ARRANGEMENTS
- RELATIONSHIP OF CODES TO OUTPUT OF THE MICRO CONTROLLER
- RC CONTROL MAPPING
- CONFIGURING THE IMU SENSORS
- UNDERSTANDING PROPELLERS, MOTORS AND MOTOR SPEED CONTROLLERS
- SETTING UP BATTERY SENSOR
- UNDERSTANDING BATTERY USAGE AND SAFETY
- PID TUNING FOR FLIGHT
- PID FOR STABILIZE MODE
- CONFIGURING TELEMETRY
- LEARN TO FLY BASICS
- PID FOR ALTITUDE HOLD
- **CONFIGURING GPS (ADVANCE LEVEL) 2560 MEGA REQUIRED**
- **PID FOR GPS MODES (ADVANCE LEVEL) 2560 MEGA REQUIRED**
- **GPS NAVIGATION (ADVANCE LEVEL) 2560 MEGA REQUIRED**

Anatomy of a DIY Drone



SYNERDUINO KWAD SHIELD KIT



Board Capabilities

Features / Flight Modes	UNO 328 (SMD)	Mega 2560	Mega 2560 + GPS
Heading hold	X	X	X
Altitude Hold	X	X	X
Headless / Course Lock	X	X	X
Telemetry	X	X	X
Payload Trigger		X	X
Camera Gimbal		X	X
RTH			X
Position Hold			X
Missions			X

Tools needed

Screwdriver / Hex set



Cutter Knife



Tape Double sided and Electrical



Zip Ties



Soldering Set



PVA Glue



Thread Lock , Purple



Tools Needed

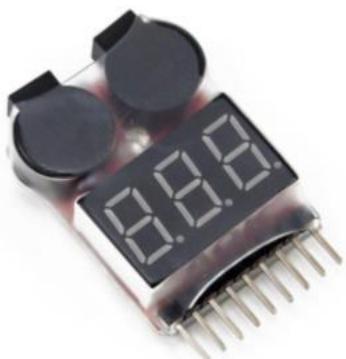
Lipo Charger 5A



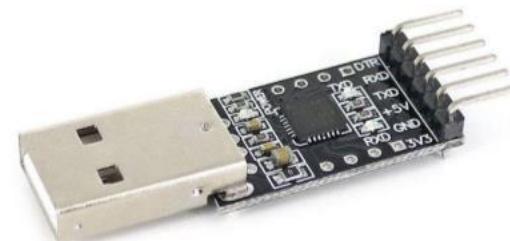
Lipo Battery 3s 1300mah



Voltage alarm

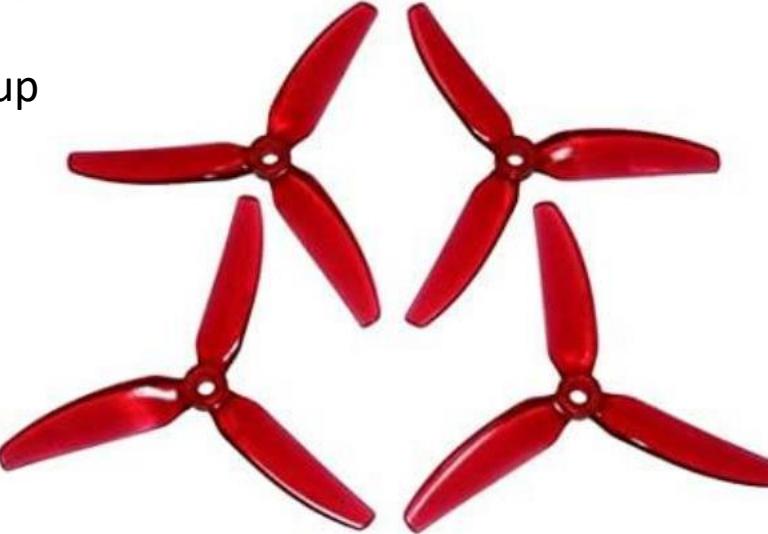


USB TTL FTDI



Hardware

Synerduino Kit + 250mm frame setup



5x4.5 3 or 5x4 3 bladed propellers (Stiff Plastic)

5x45x3 (5045 3) or 5x40x3 (5040 3)

Counter rotation CCW & CW



Bluetooth HC05 and GPS



XT60 Plug



Servo Wires



ESC OPTO Standard PWM or BL Heli or
BL Heli-S 20A-30A 3s-4s

ESC With BEC if you needed to run extra
servos ,sensors or accessories



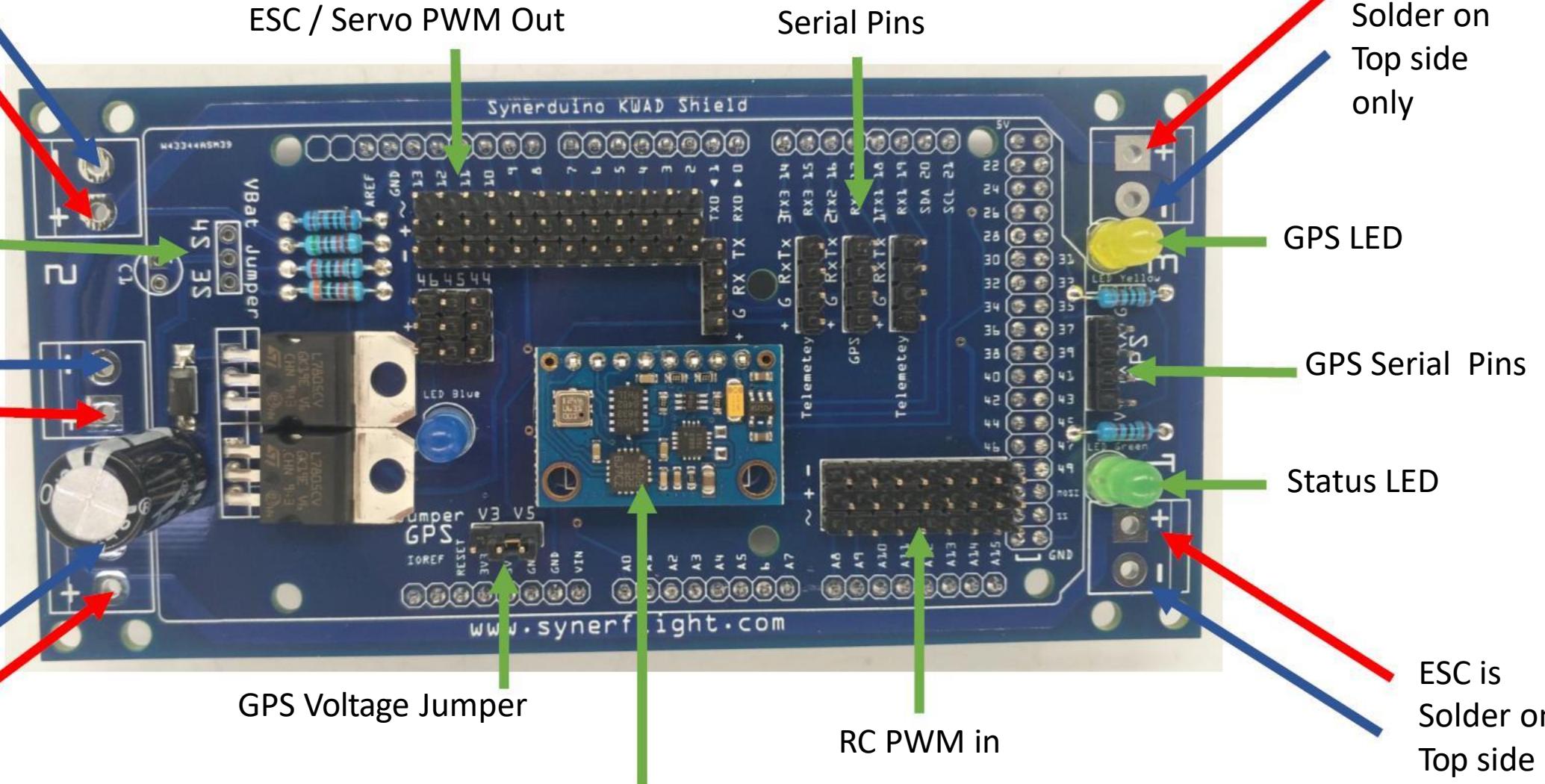
5V 3A UBEC – for those who needs to run extra
servos or sensor components

Recommend 2200 - 2300 Size 2000Kv-2400KV motors those with 22mm Stator size average rate of 1000g thrust

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



ESC is
Solder on
Top side
only

ESC is
Solder on
Top side
only

GPS Serial Pins

Status LED

ESC is
Solder on
Top side
only

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

Set jumper to the
Battery Cell
count (Soldered) 3s
– 4s

Power
input
3s 11.1V
4s 14.8V

GPS Voltage Jumper

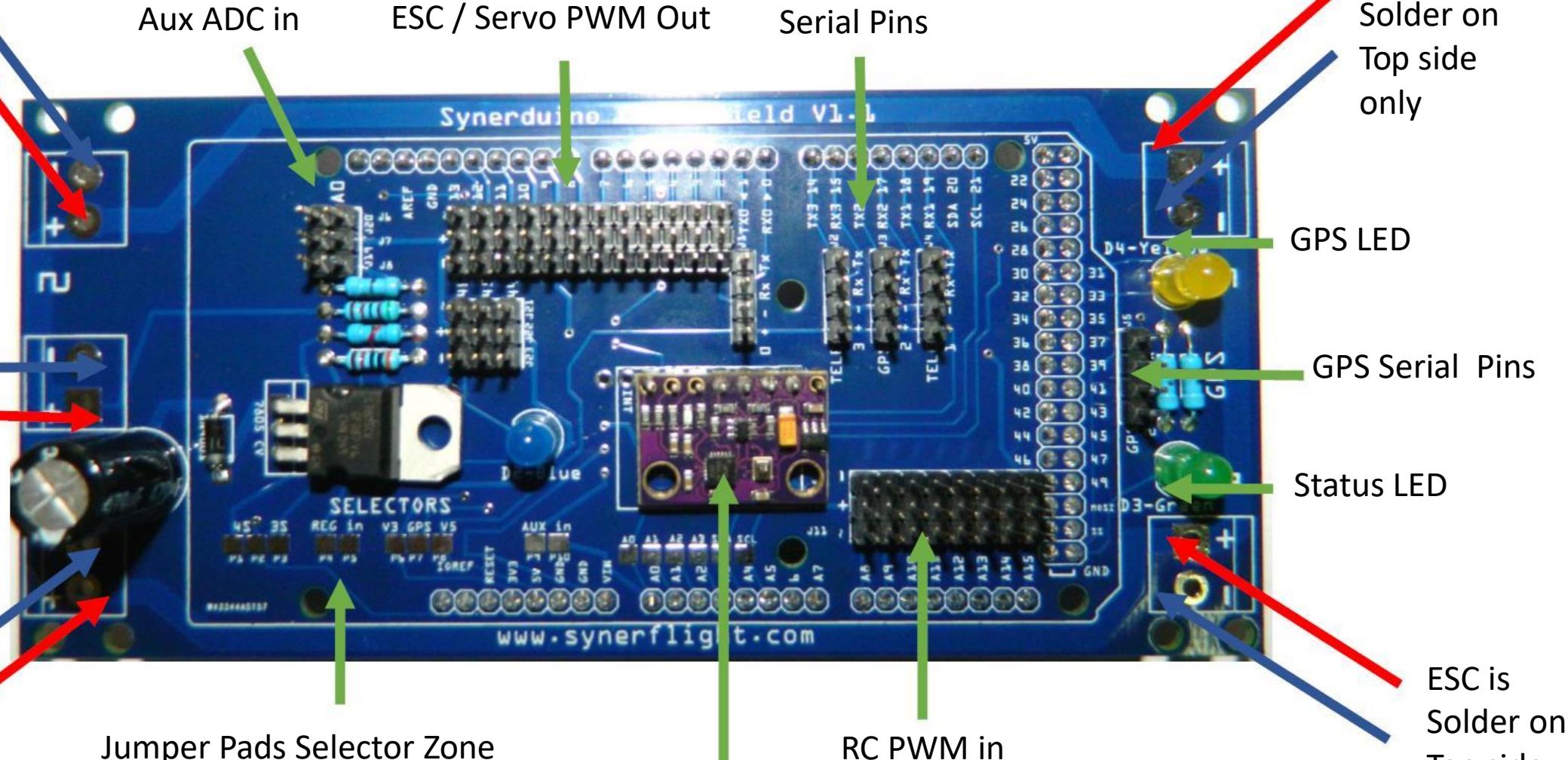
RC PWM in

For improve performance IMU must be protected from the Environment

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



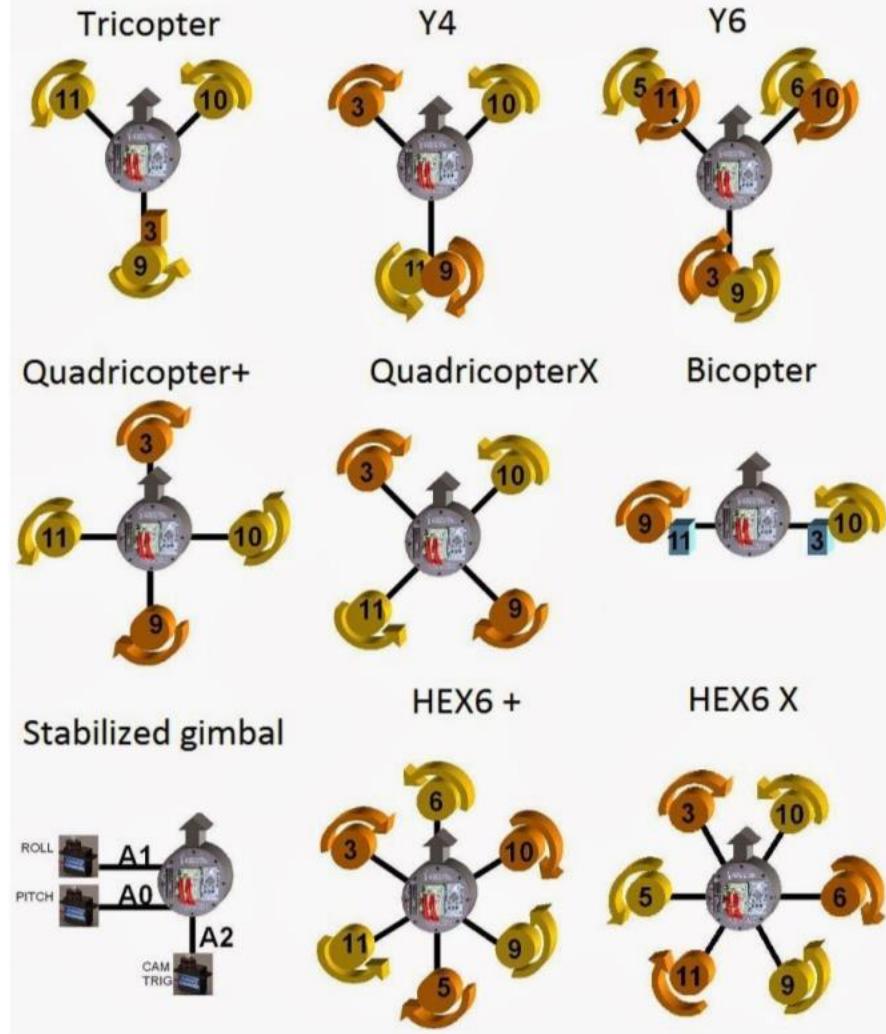
ESC is
Solder on
Top side
only

For improve performance IMU must be protected from the Environment

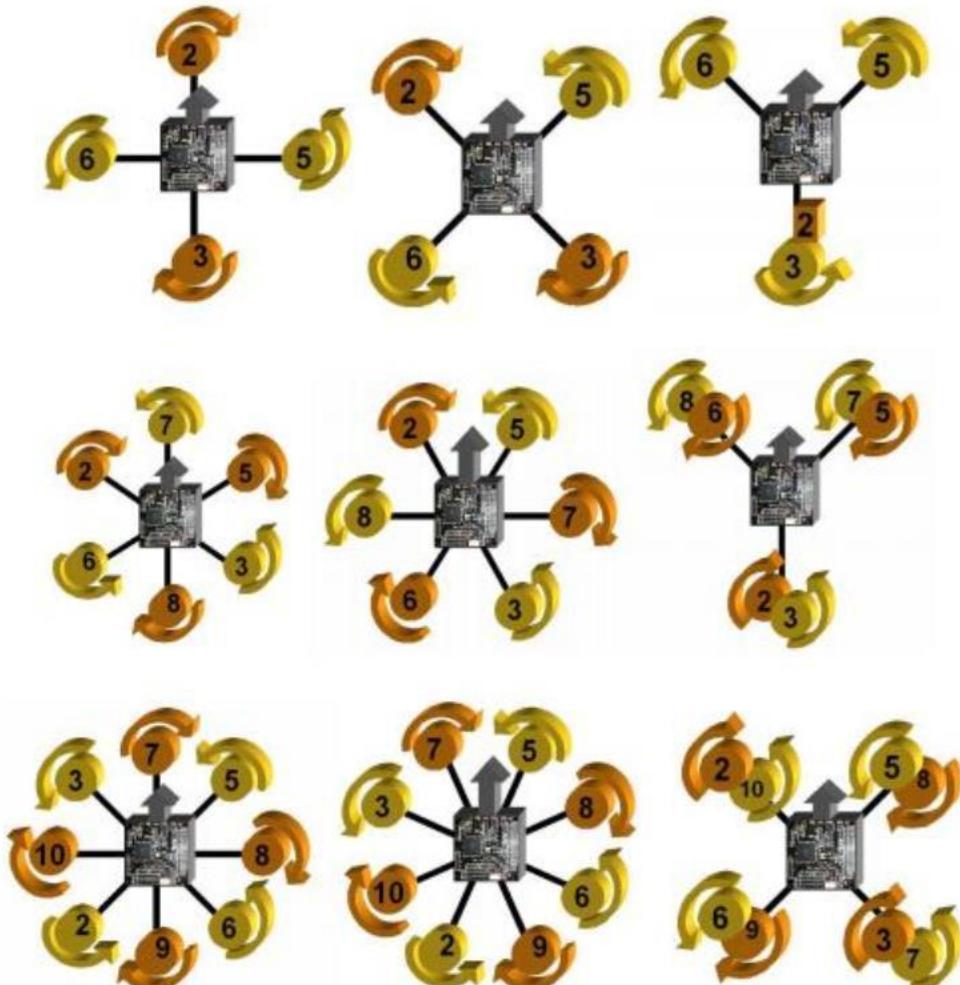
IMU : MPU-9250 & BMP280

ESC is
Solder on
Top side
only

PWM Pins arrangement D2-D10 / PWM Output



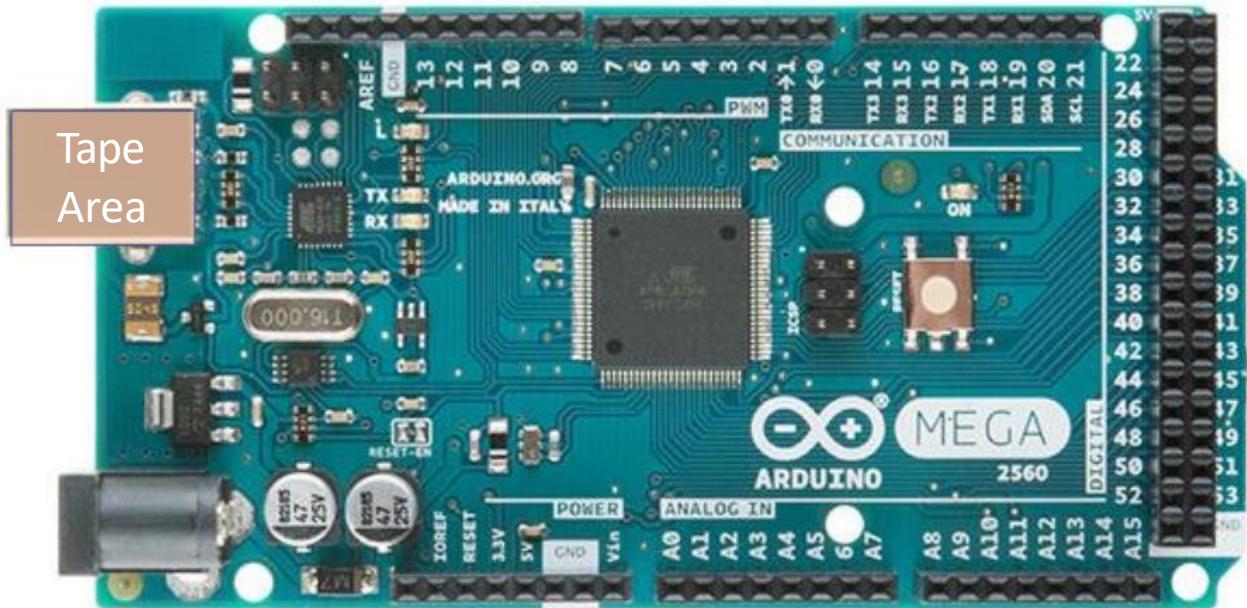
UNO 328



MEGA 2560

Arduino Board Preparation

Ensure insulation from the Arduino board add tape on these areas



2560 MEGA

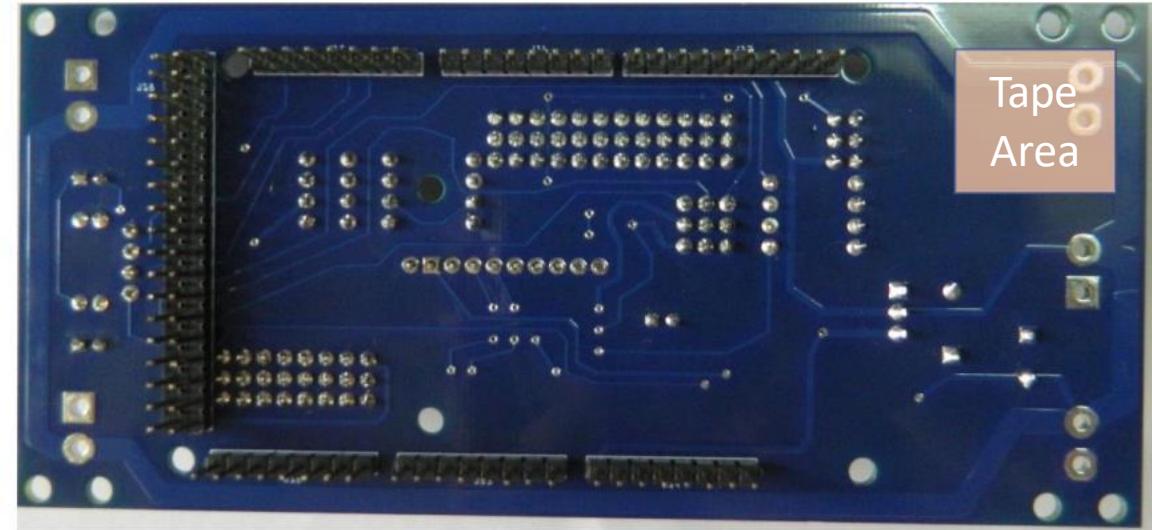
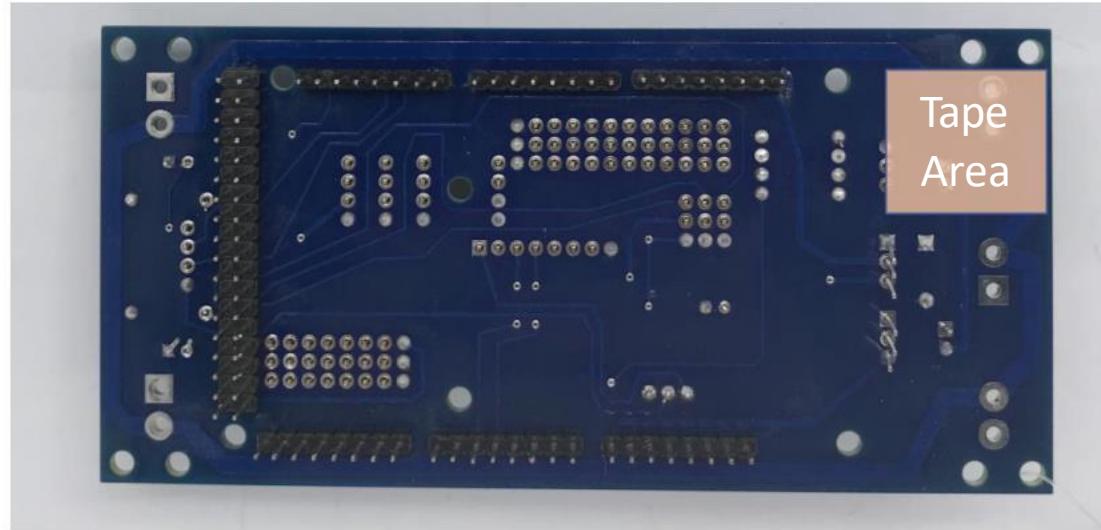


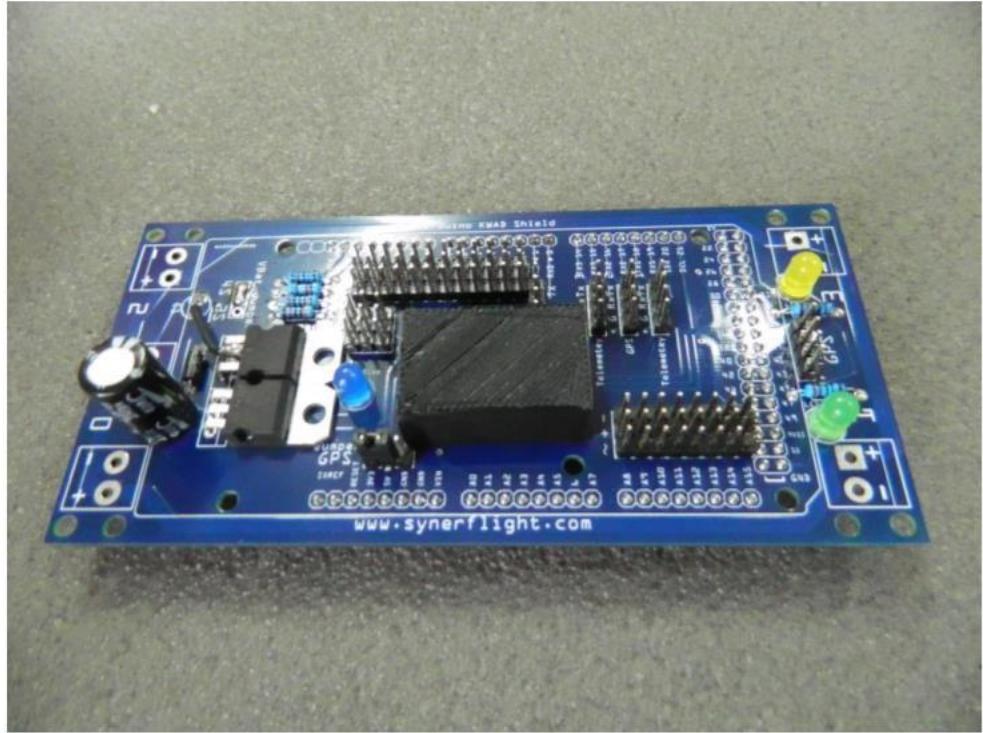
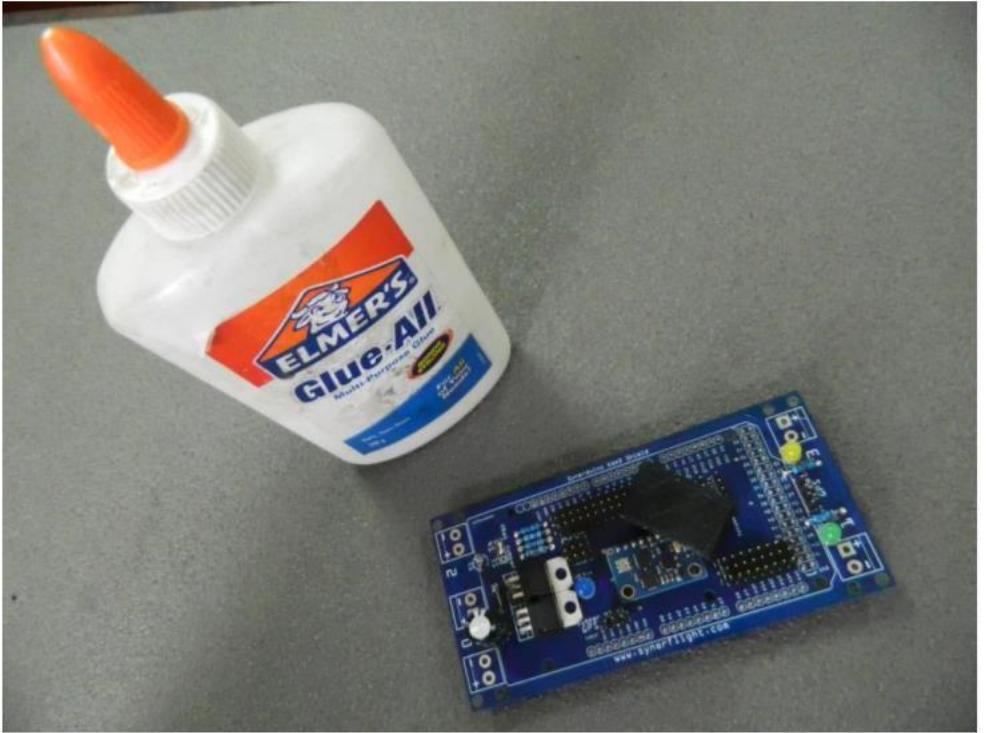
SMD UNO 328

Synerduino Kwad Shield Preparation

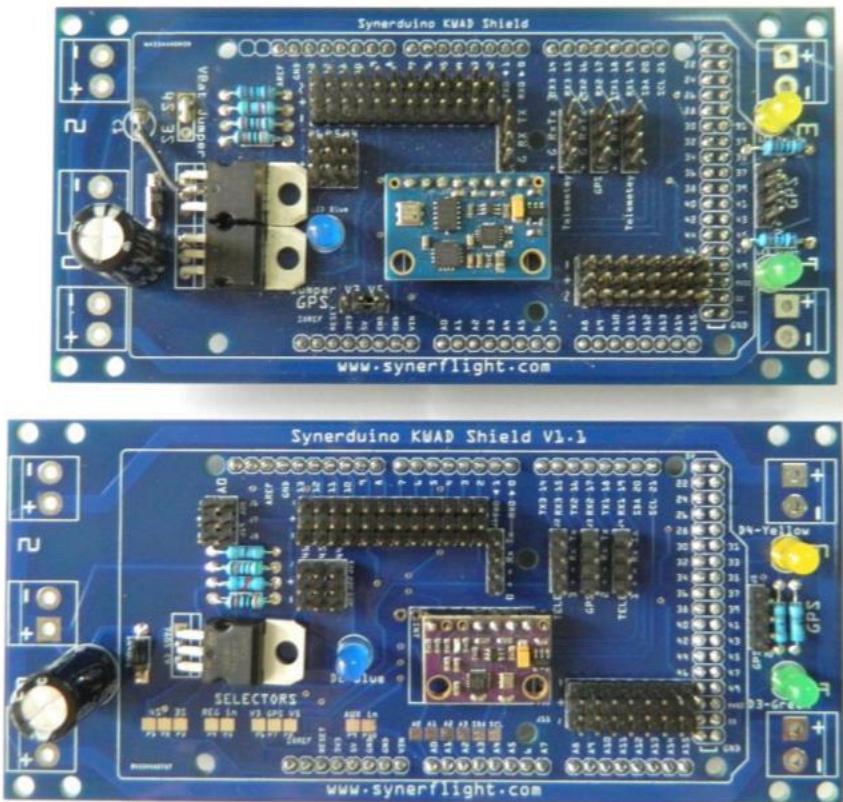
Ensure insulation from the Arduino board add tape on these areas

BETA GY801 & V1.1 GY91





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



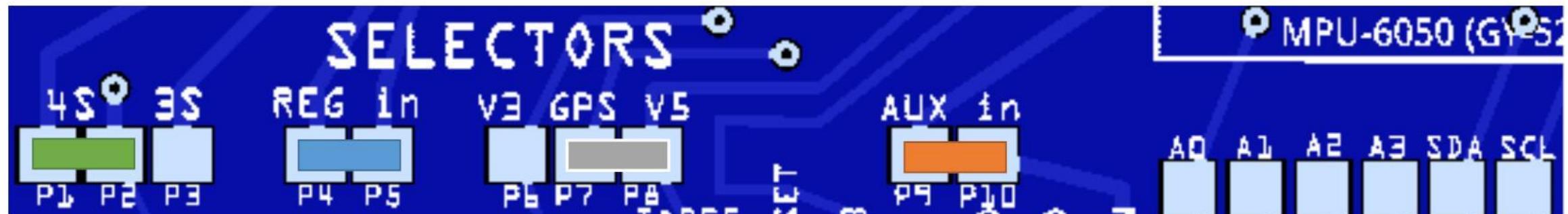
Synerduino Kwad shield V1.1

Update is 1cm longer board to clear the USB port from the solder pin

Synerduino Kwad Shield V1.1 (GY91) POWER Selector

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

Analog 0 pin Auxin / Battery monitor

A0 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

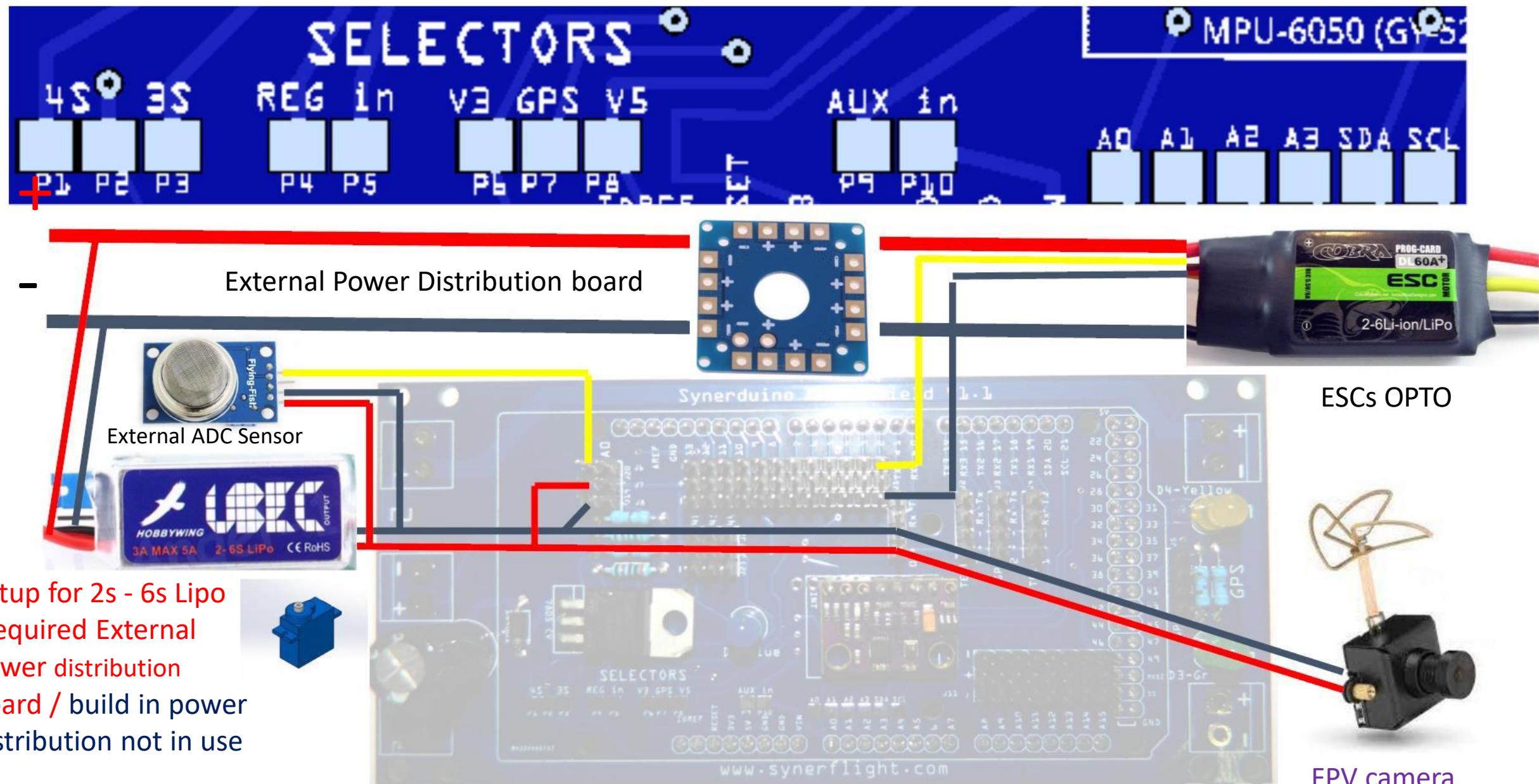
Reg In – short the Pads for using regulator to power and build in power distributor the synerduino and Arduino board

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

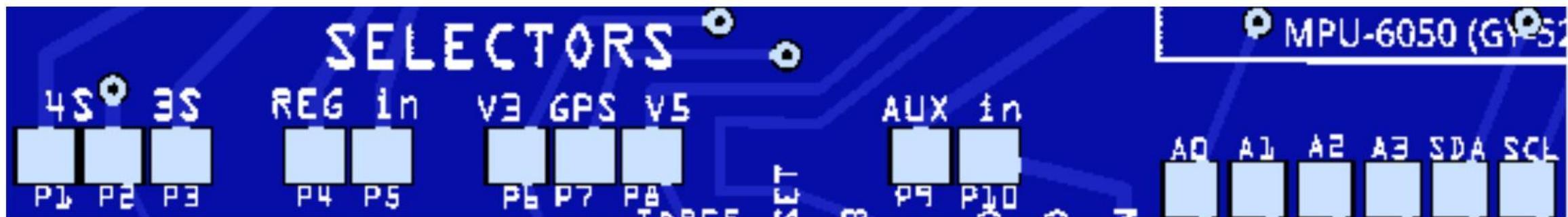
Aux in- **leave it open** for utilizing the A0 Pins for External ADC sensors / Short the Pads to use build in battery monitoring . Cell Selector must be set to 4s or 3s

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

Default - A0 External ADC sensor , Require 5v UBEC applied to the 5V Serial pins & for OPTO ESC



Default - A0 External ADC sensor , Require 5v UBEC applied to the 5V Serial pins & for OPTO ESC



Bypass Regulator method also use for Servo , FPV and External sensor intended for Current intensive setups

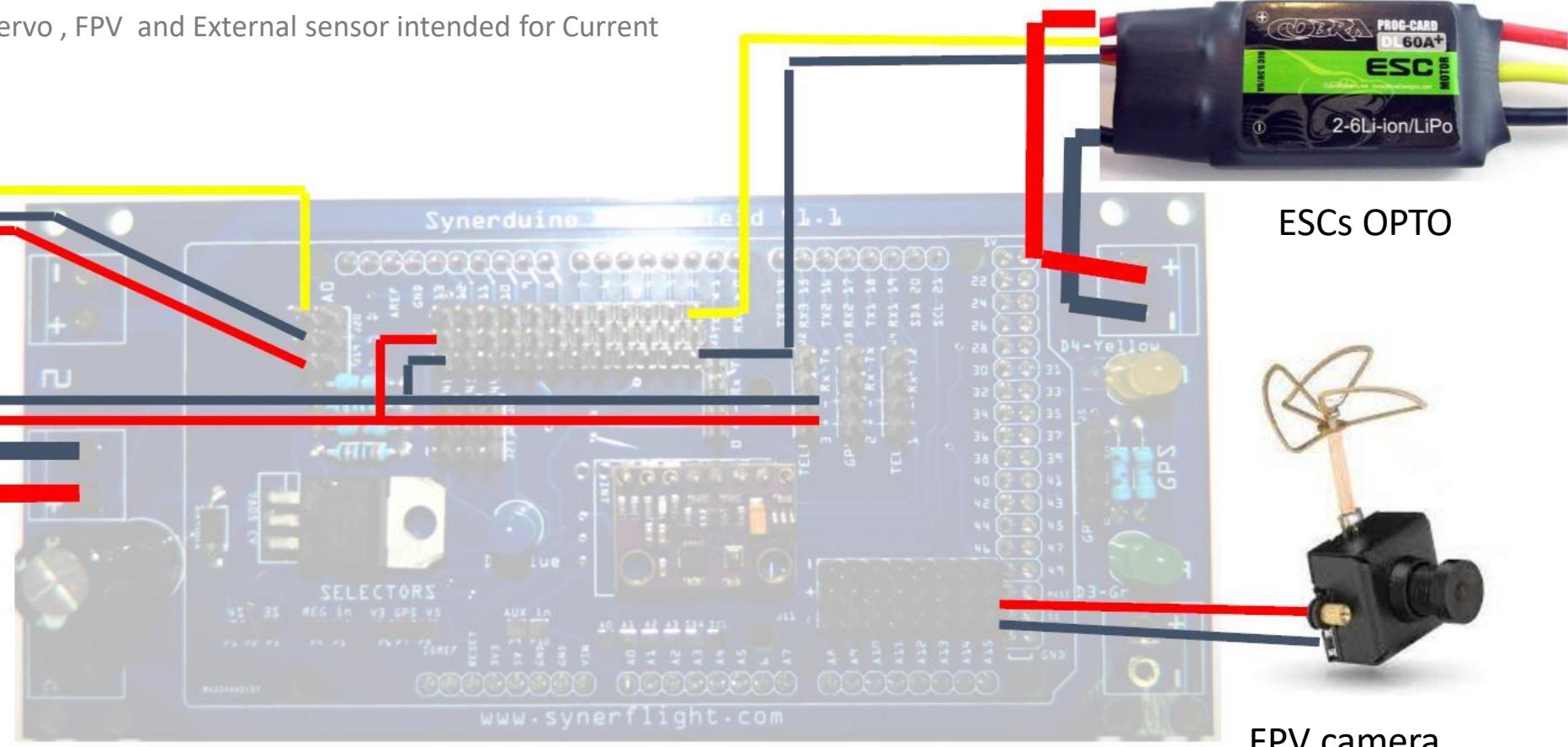
External ADC Sensor



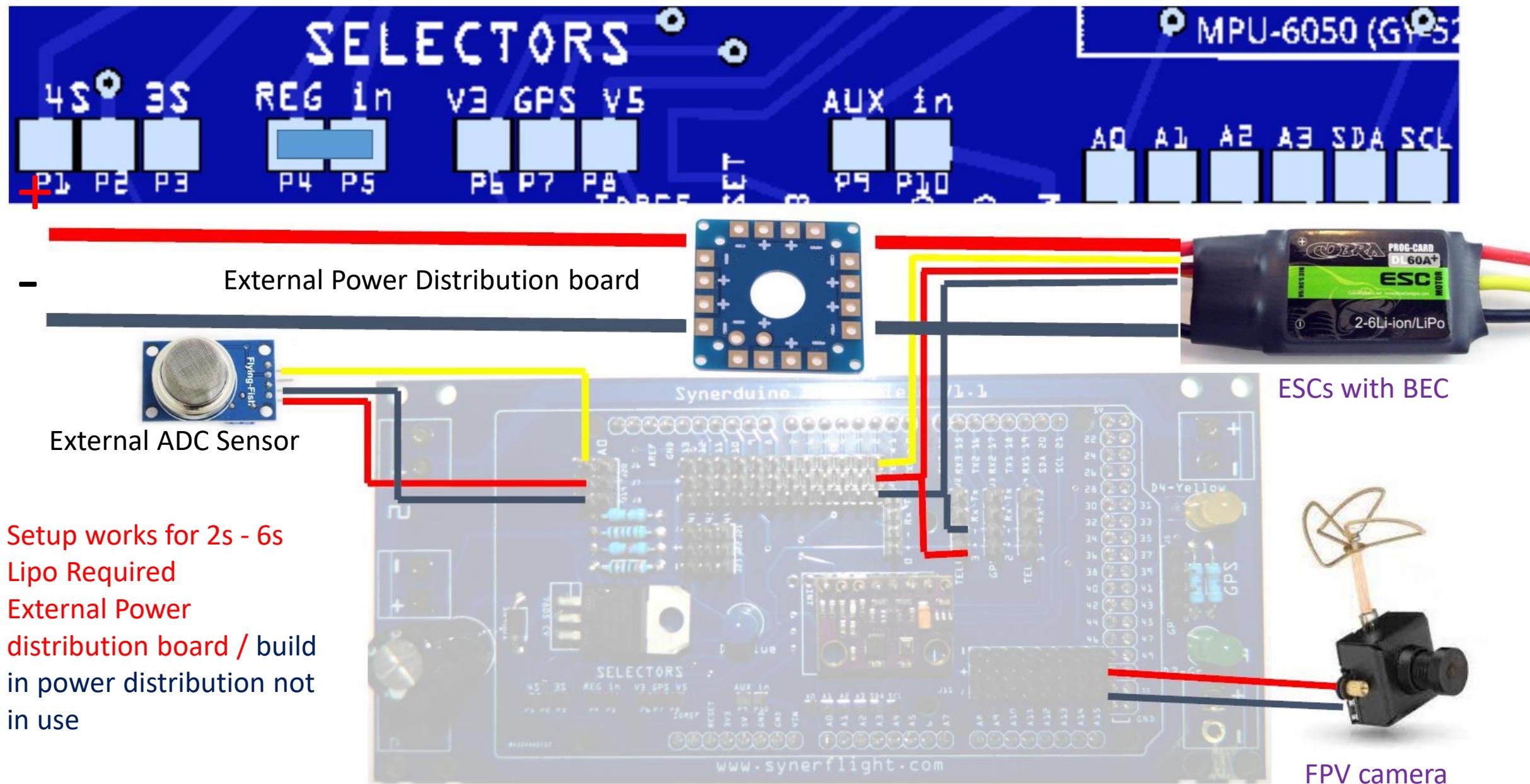
5V BEC



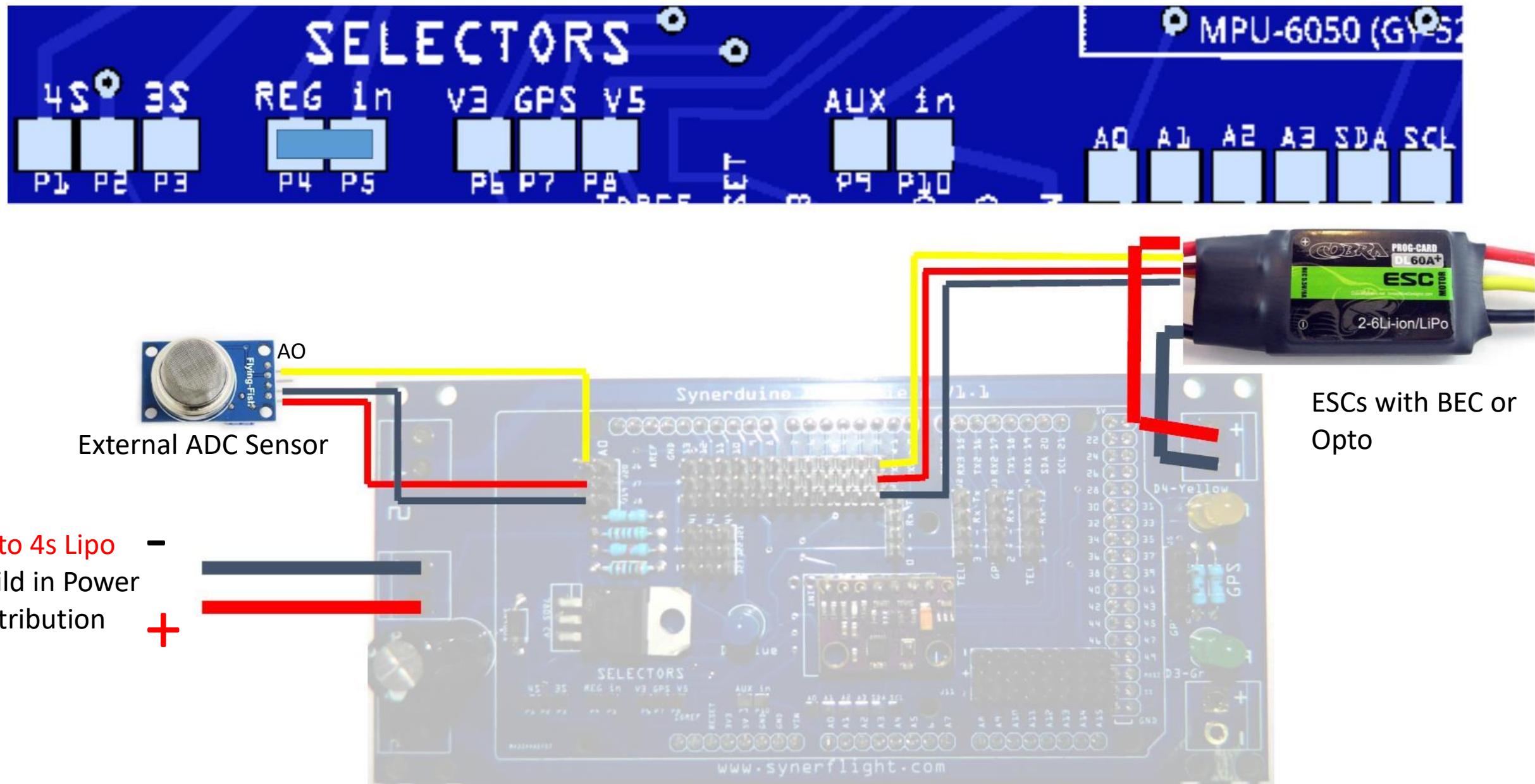
+ Setup for 2s - 4s Lipo
Required External
Power distribution
board / build in power
distribution not in use



Reg in only - A0 External ADC sensor , Require ESC with UBEC applied to the 5V PWM pins

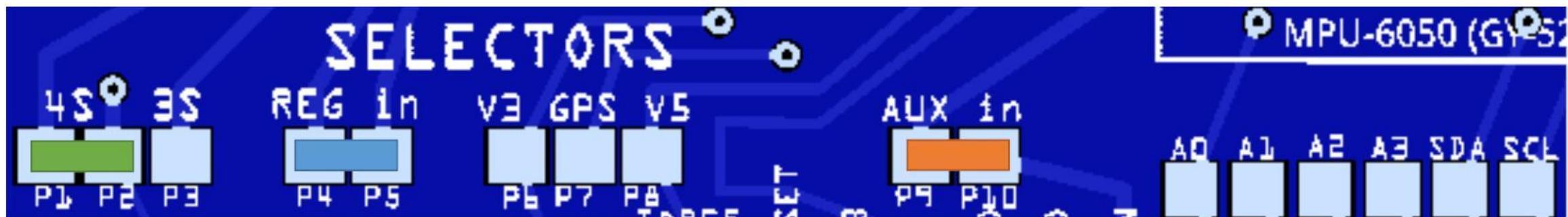


Reg in only - A0 External ADC sensor , ESC BEC or OPTO applied to the 5V PWM pins

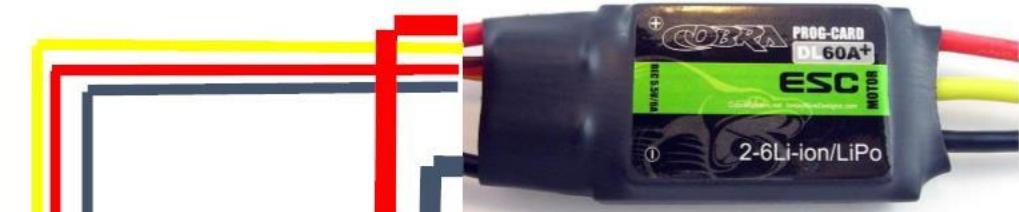


Recommended setup for beginner

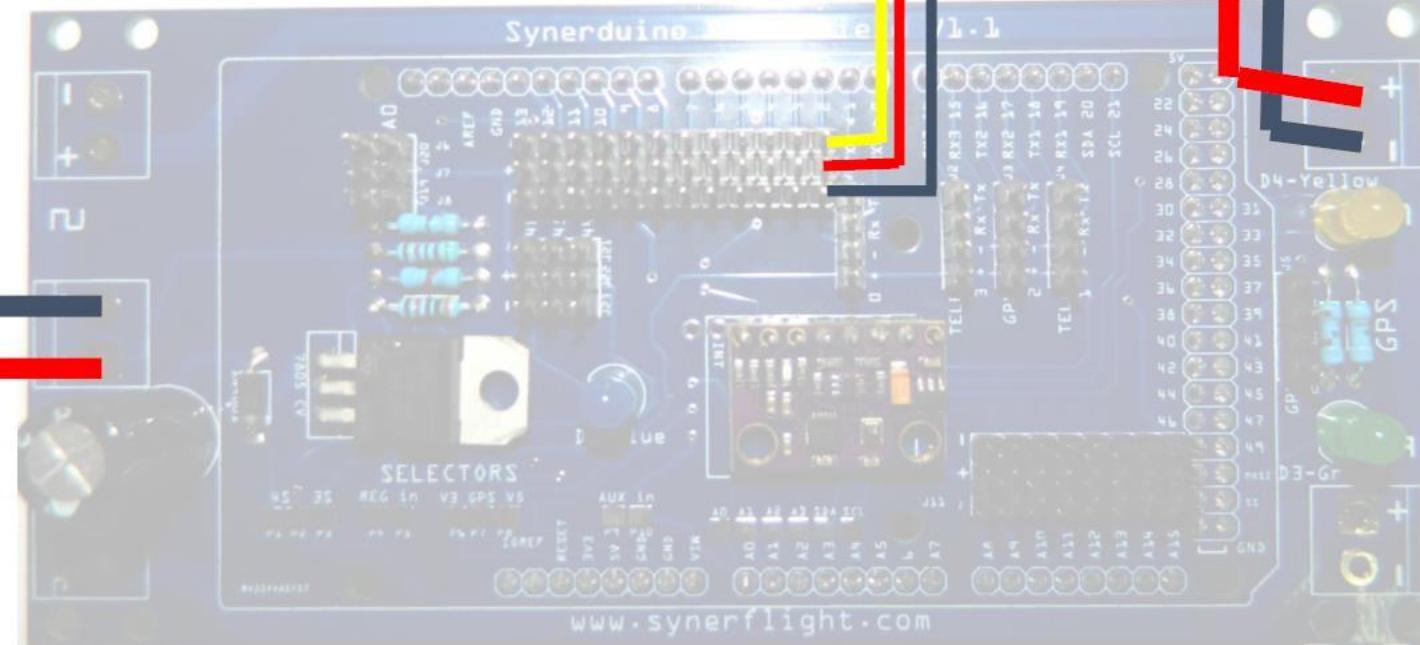
Reg & Vbat - A0 as Battery voltage monitor , ESC BEC or OPTO applied to the 5V PWM pins



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



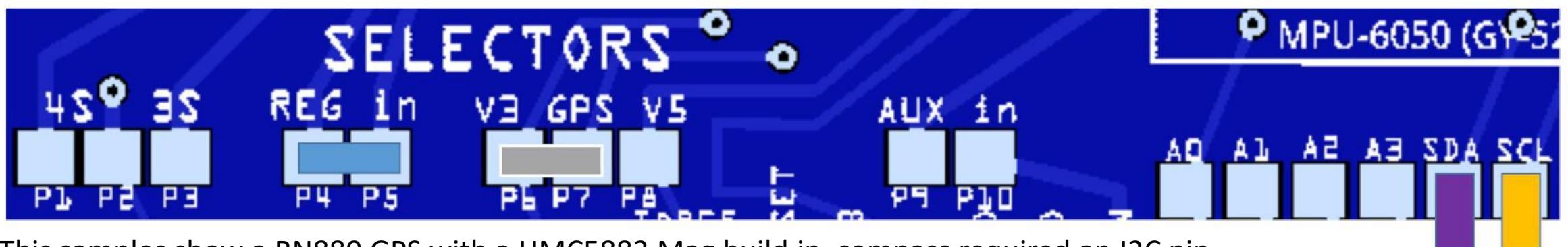
ESCs with BEC or Opto



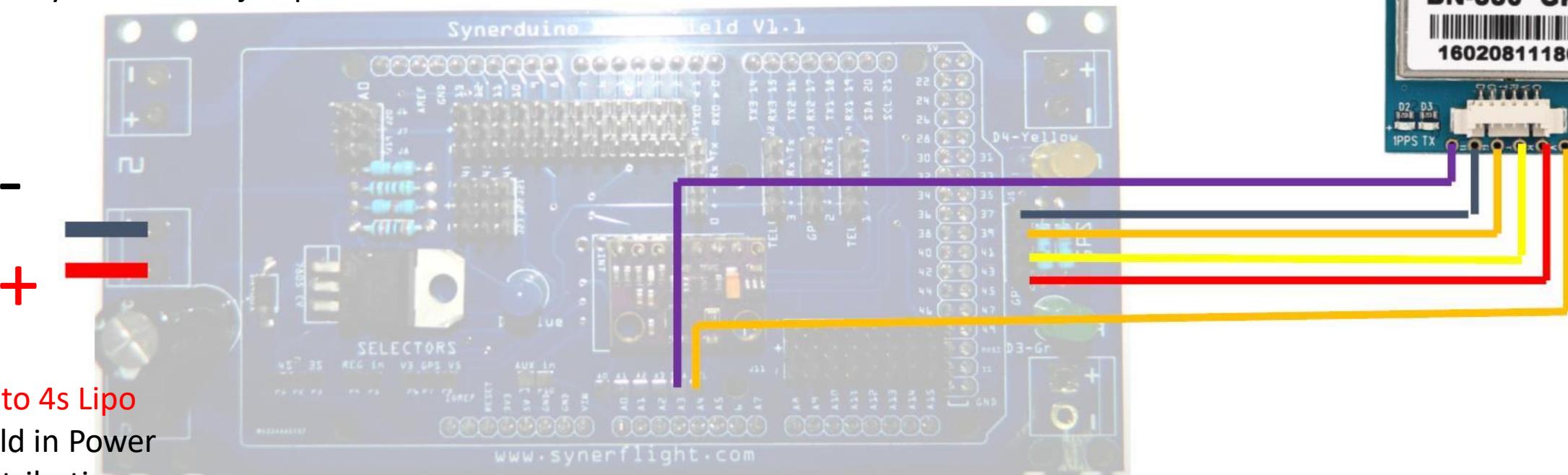
2s to 4s Lipo -
Build in Power distribution +

Recommended setup for beginner

Reg & External Sensors

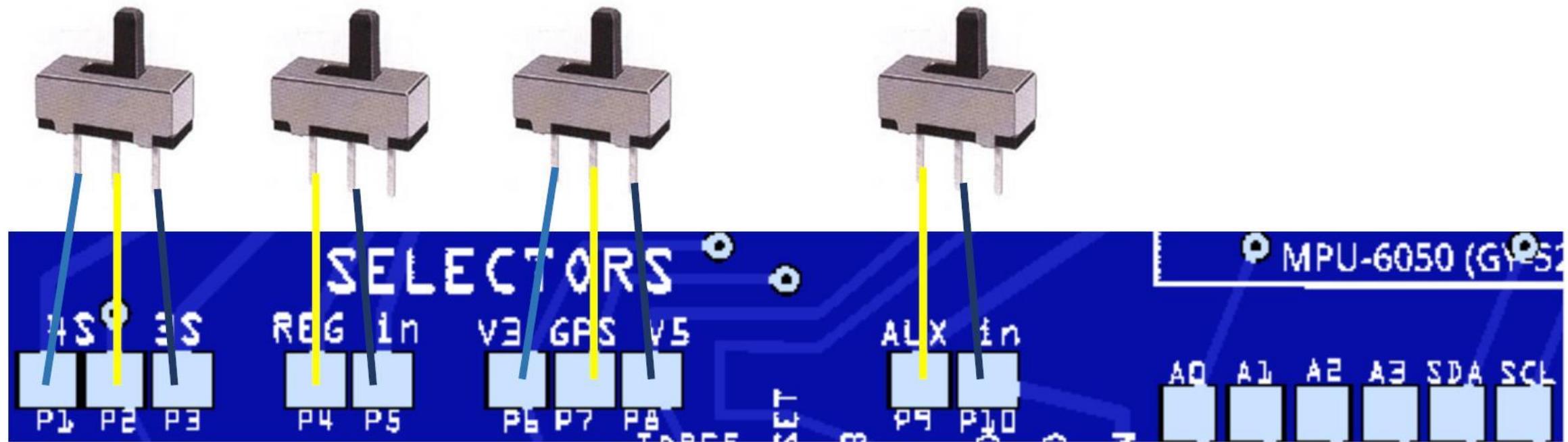


This samples show a BN880 GPS with a HMC5883 Mag build in compass required an I2C pin connection this works of all other I2C sensors (pls ensure the address doesn't conflict with the IMU as found in Sensors.cpp) Note: other than the GPS build in sensors might require 3V you may need to set jumper to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

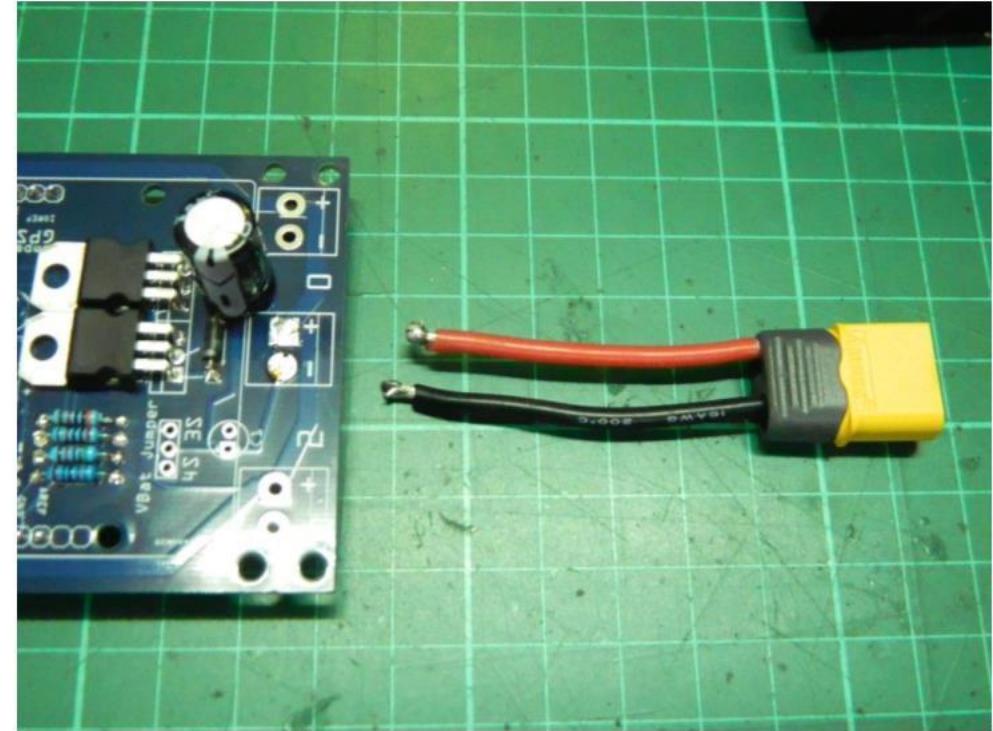
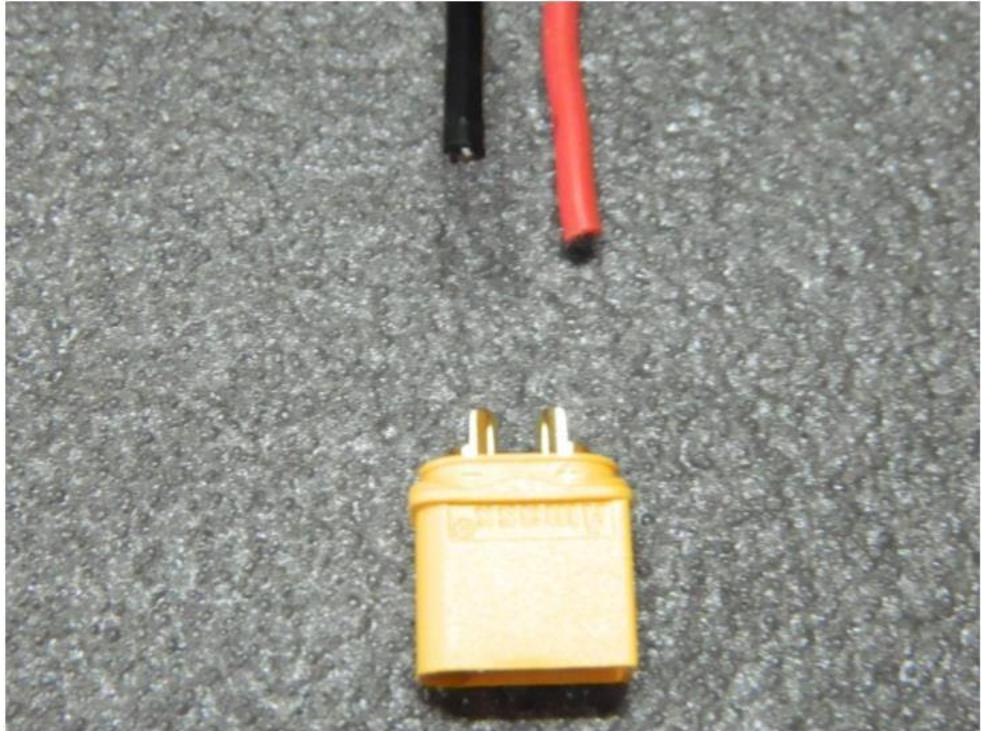


2s to 4s Lipo
Build in Power distribution

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

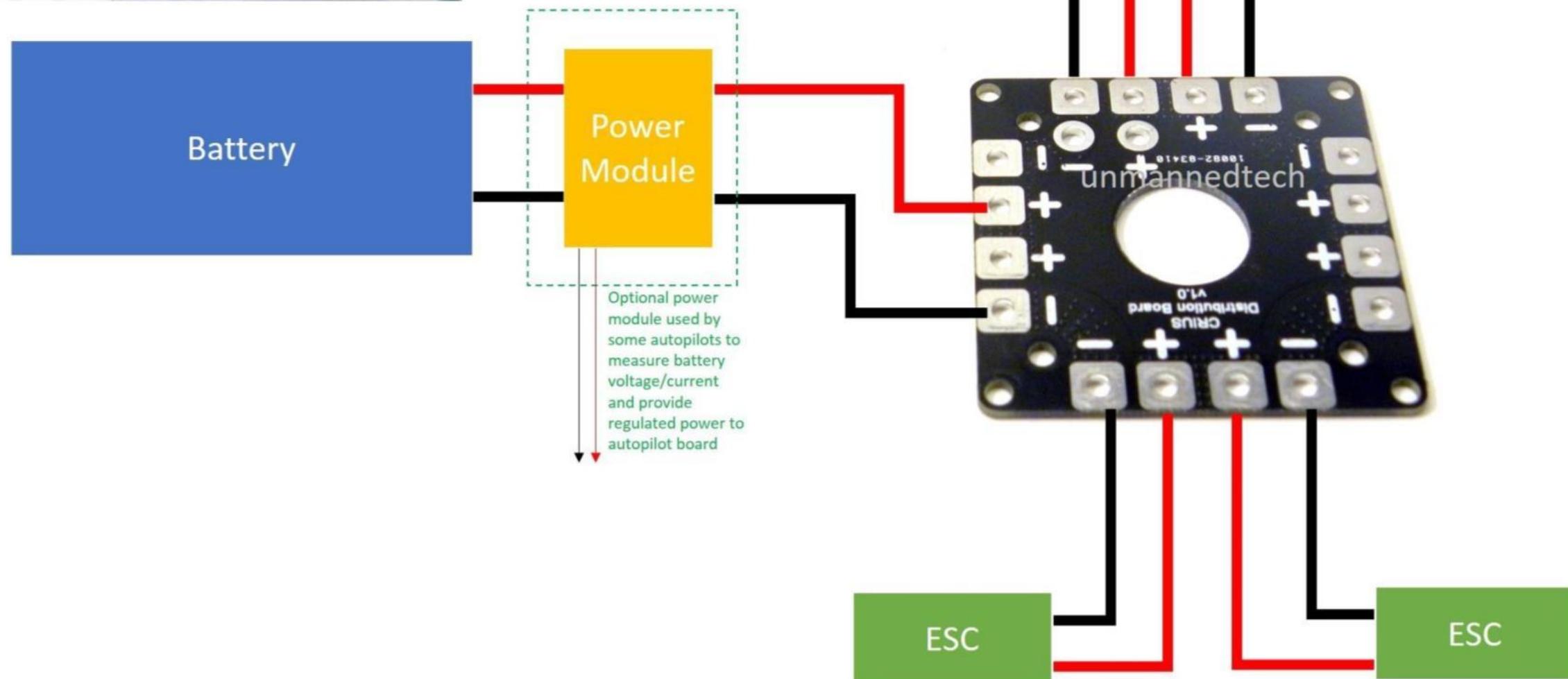


For those who would use the build in battery monitoring
circuit upto 4s lipo **ensure the Cell Count and Aux in is jumped**
before powering up



Please check the polarity of the XT60Plugs 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



Electronic Speed Controller



ESC size i.e. the amount of amps provided to your motor

Extra info such as BEC and what battery the ESC can support



Motor type	The voltage (V)	Paddle size	current (A)	thrust (G)	power (W)	efficiency (G/W)	speed (RPM)
RS2205-2300KV	12	HQ5045 BN	1	62	12.00	5.17	6400
			3	162	36.00	4.50	10080
			5	236	60.00	3.93	12070
			7	311	84.00	3.70	13730
			9.1	374	109.20	3.42	15100
			11	439	132.00	3.33	16320
			13	490	156.00	3.14	17350
			15.3	548	183.60	2.98	18350
			17.3	611	207.60	2.94	19210
			20.7	712	248.40	2.87	20080

To start from the very beginning, ESC are of course electronic speed controller and these are the little devices that control the speed of your brushless motors. They come with different amperage and the size of ESC that you need depends on your multi-rotor setup

The Amperage should match the Motors Amperage Draw as found on the Tech specs of the Motor

2nd is the Type of ESC

- OPTO ESC is isolated meaning only two lines of the RC input are visible the PWM and Ground
- BEC ESC on the other hand includes the 5V in to power your electronics and it would be stated on the label /Data sheet of the ESC on how many Amps it would support in powering your RC electronics (Receiver, Flight Controller, Servos ,etc)

ESC are to be calibrated on first run as calibration process are required

Electronic Speed Controller Compatibility and Protocol

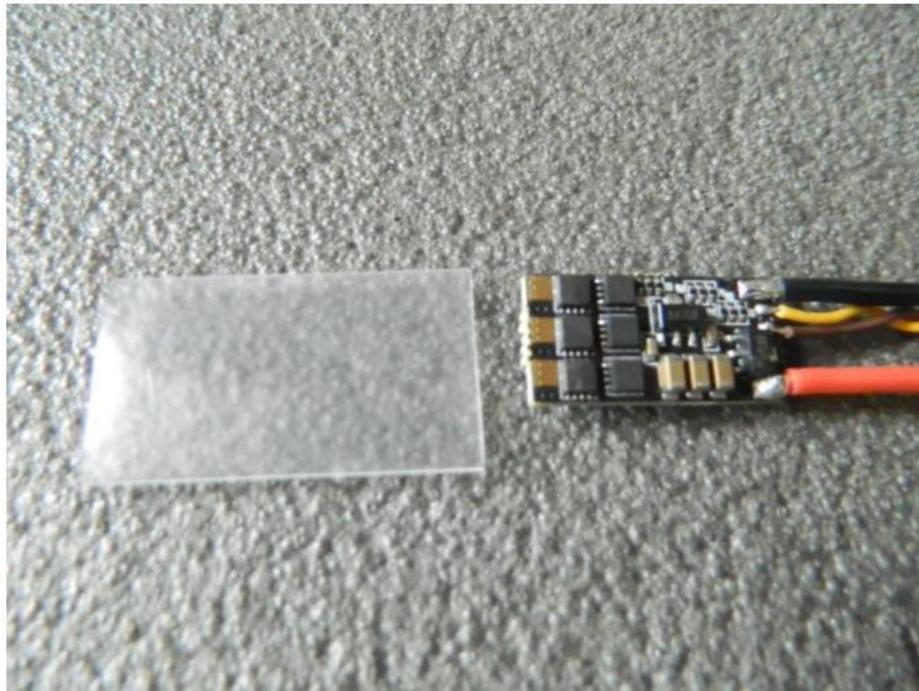
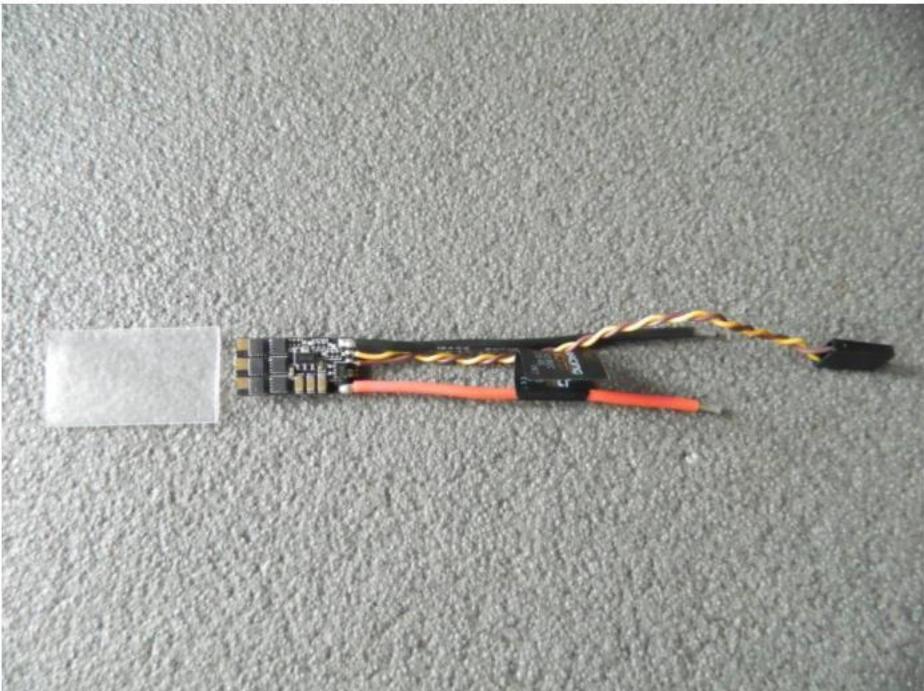
There are several types of ESC protocol and ESC firmware available for quadcopter

Because Synerduino is a shield base of the 8bit Arduino platform the speed of the PWM can be restrictive

The Synerduino would largely run off Standard PWM Protocol (1000us – 2000us)

Compatible ESC Firmware	Protocol
SimonK	Standard PWM (1000us – 2000us)
BL Heli	Standard PWM (1000us – 2000us) Oneshot125 (125us – 250us)
BL Hei-S	Standard PWM (1000us – 2000us) Oneshot125 (125us – 250us) OneShot 42

BLHeli_32 and Dshot ESC may not be compatible with the Arduino due to its speed requirements

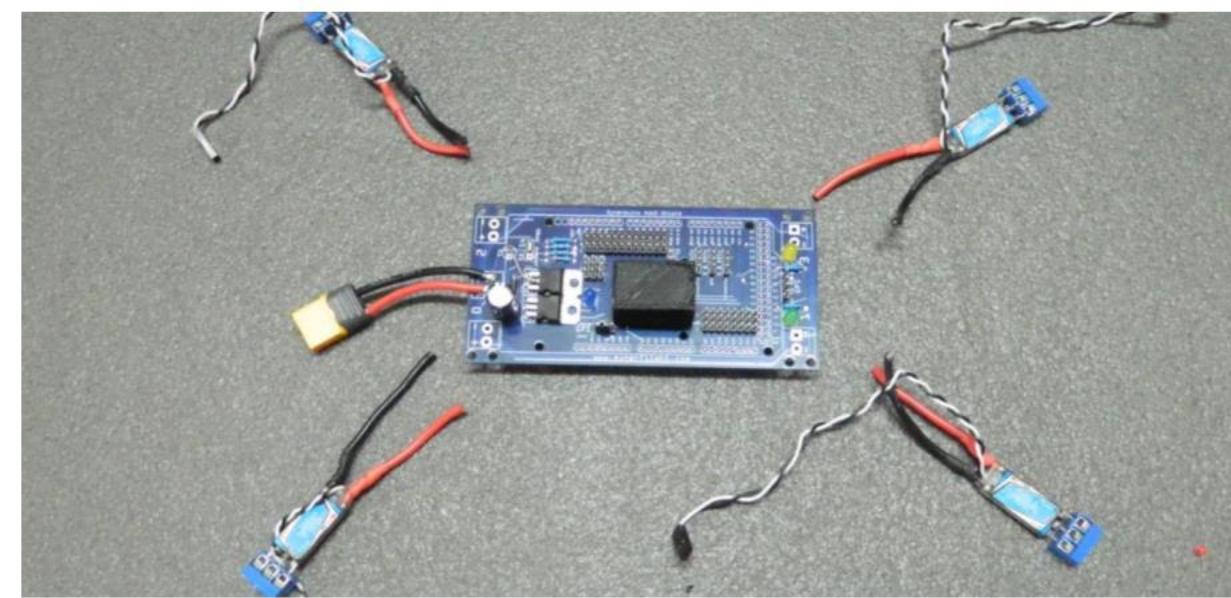
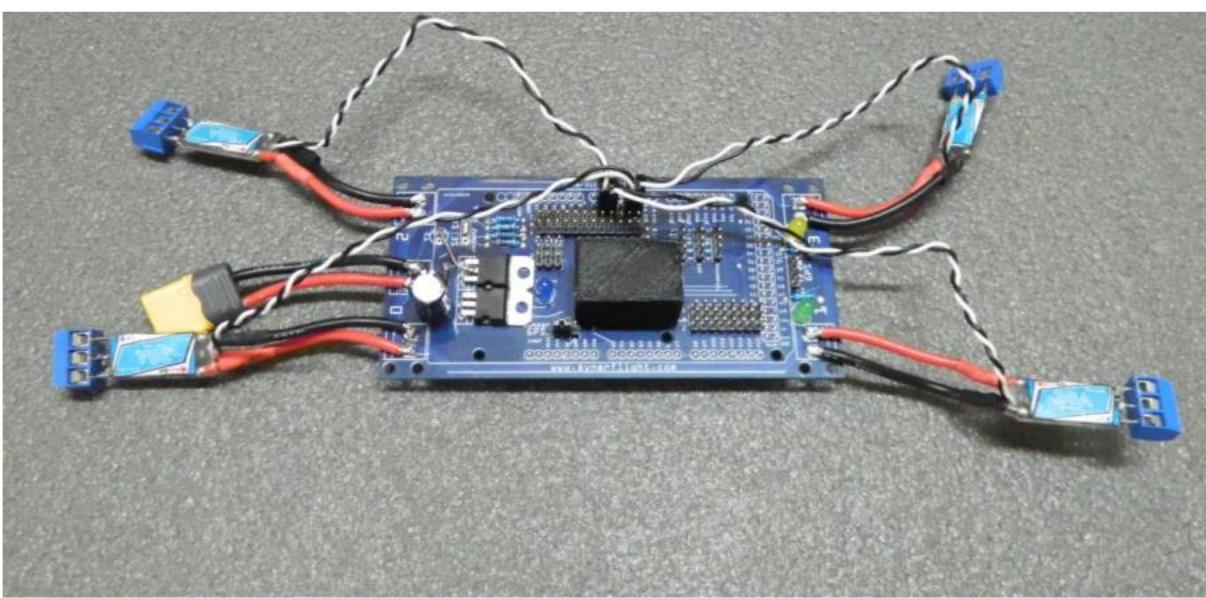
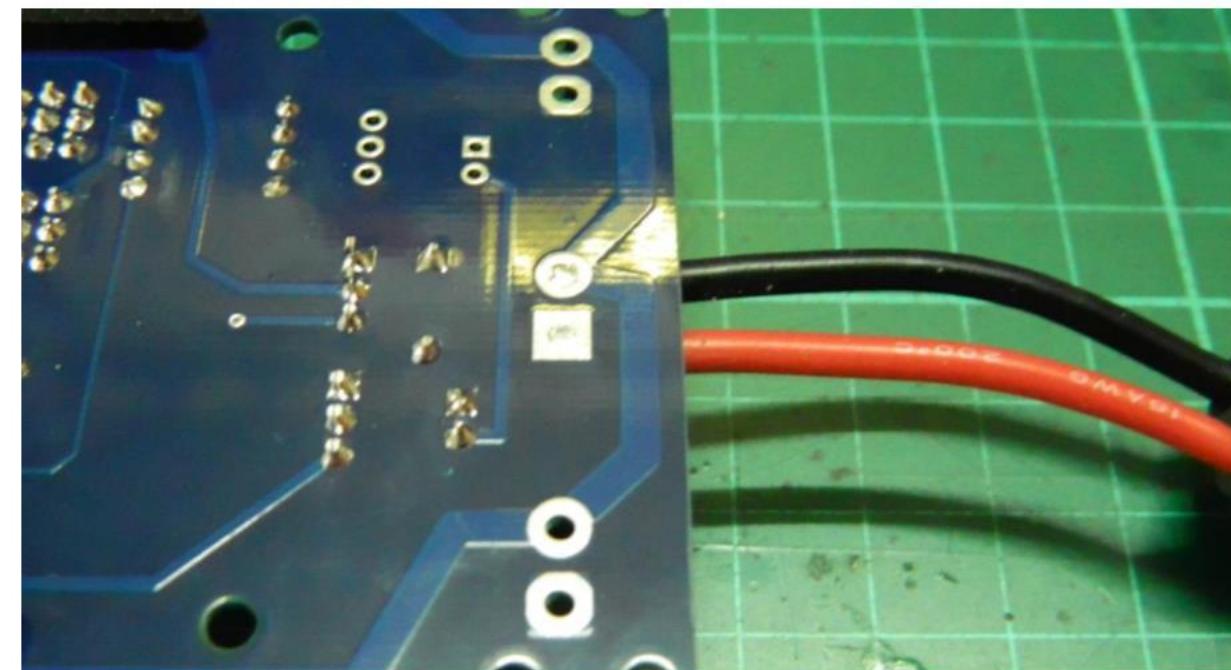
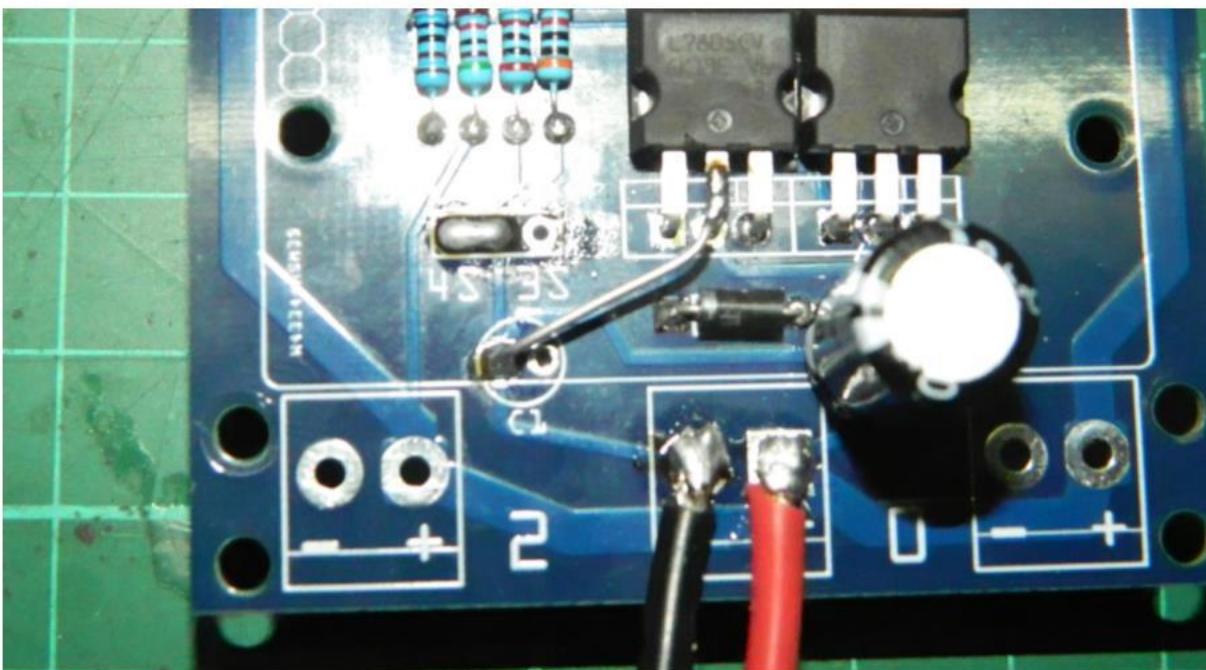


For Small Race ESCs solder the motor lines in first before applying the shrink tube



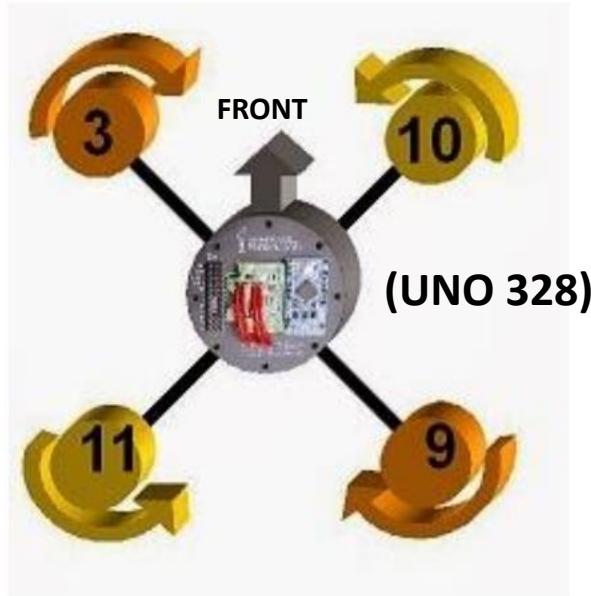
BEC “Universal Battery Elimination Circuit”

OPTO 'opto-coupled', throttle signal is coupled into the ESC is isolated



Electronic Speed Controller

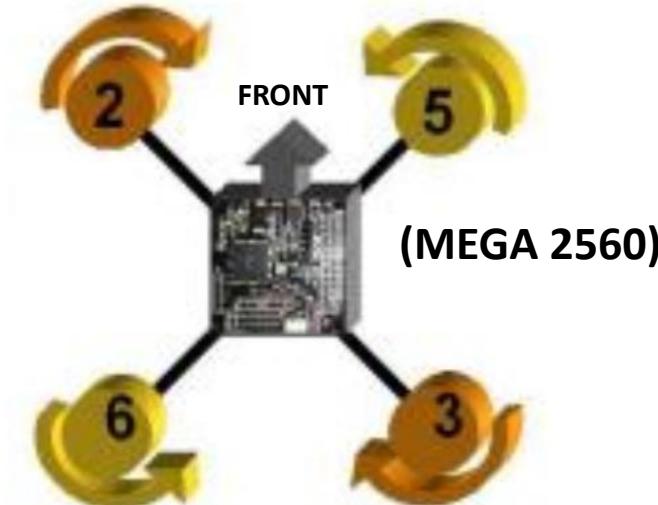
Motor [3]



Motor [2]

Motor [3]

Motor [2]



Motor [1]

Motor [0]

Motor [1]

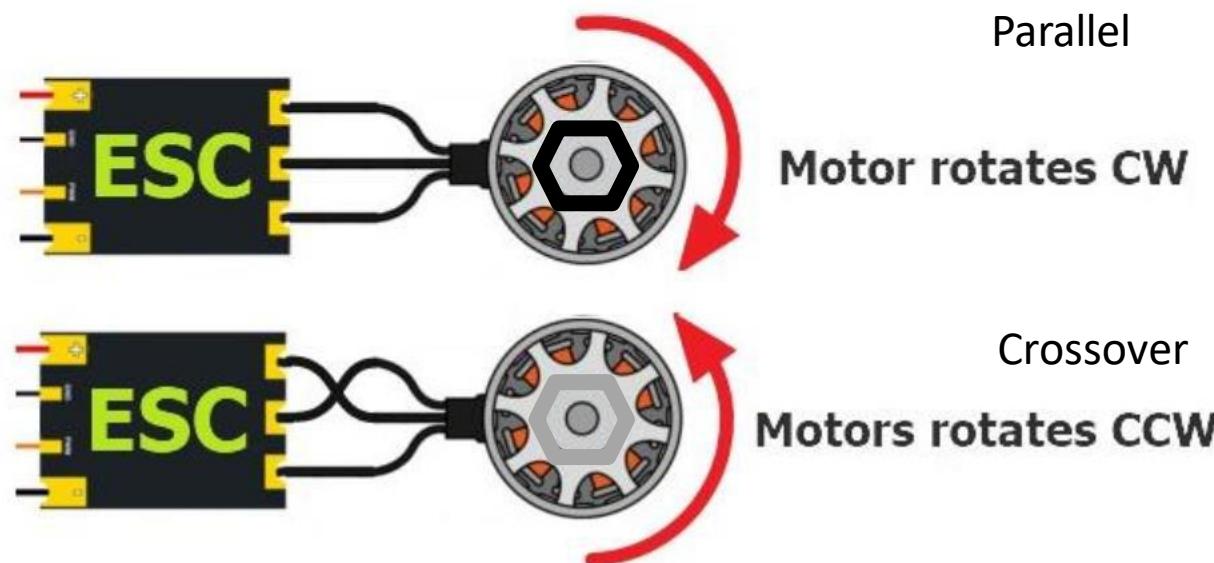
Motor [0]

Note : you can pre solder the motor to the board and check for rotation before installing the propeller to insure all motor rotations are correct

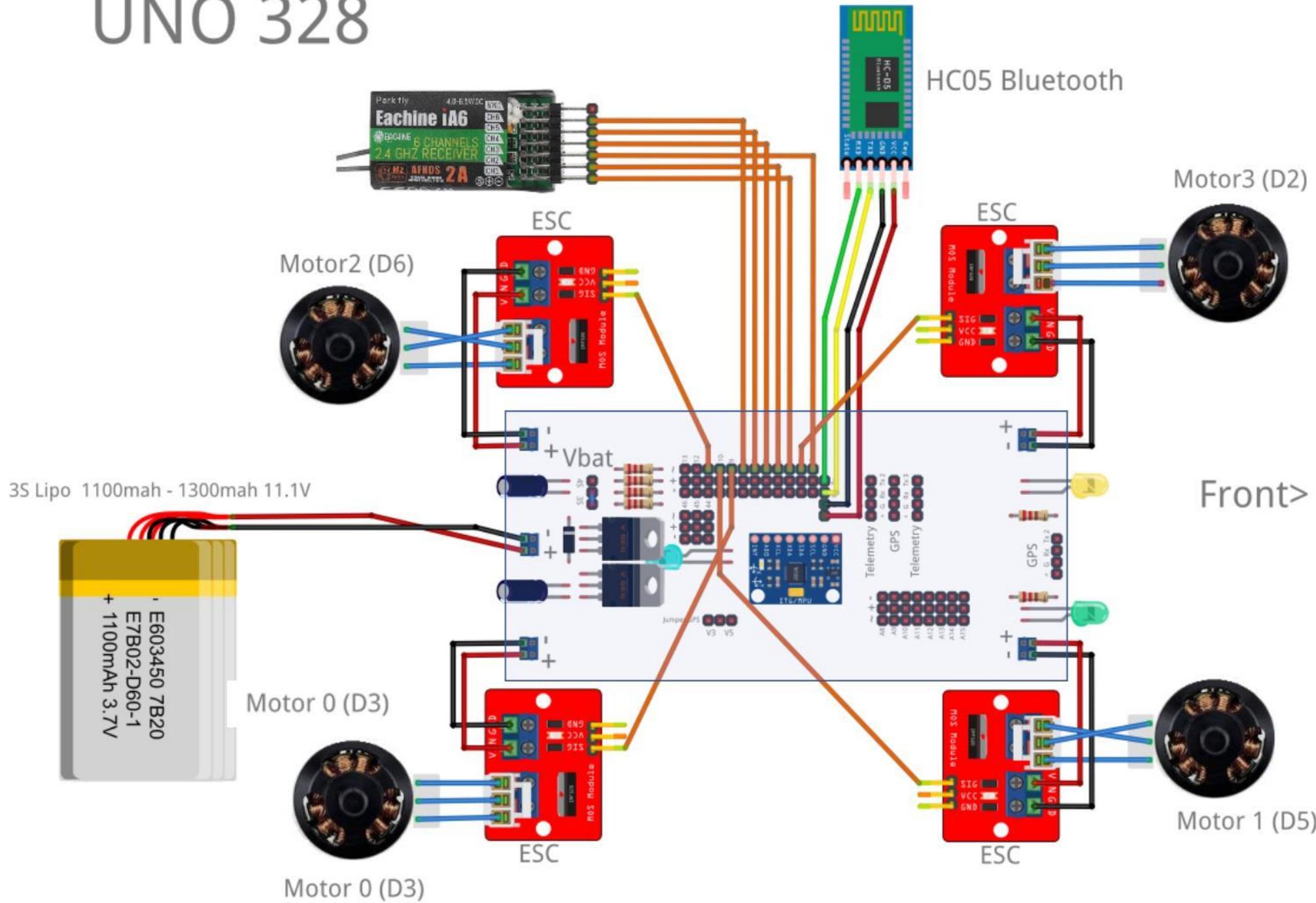
Note : on some brands of motor they may came in two different prop nuts color (Known as self tightening nuts)

Black for Clockwise Motor

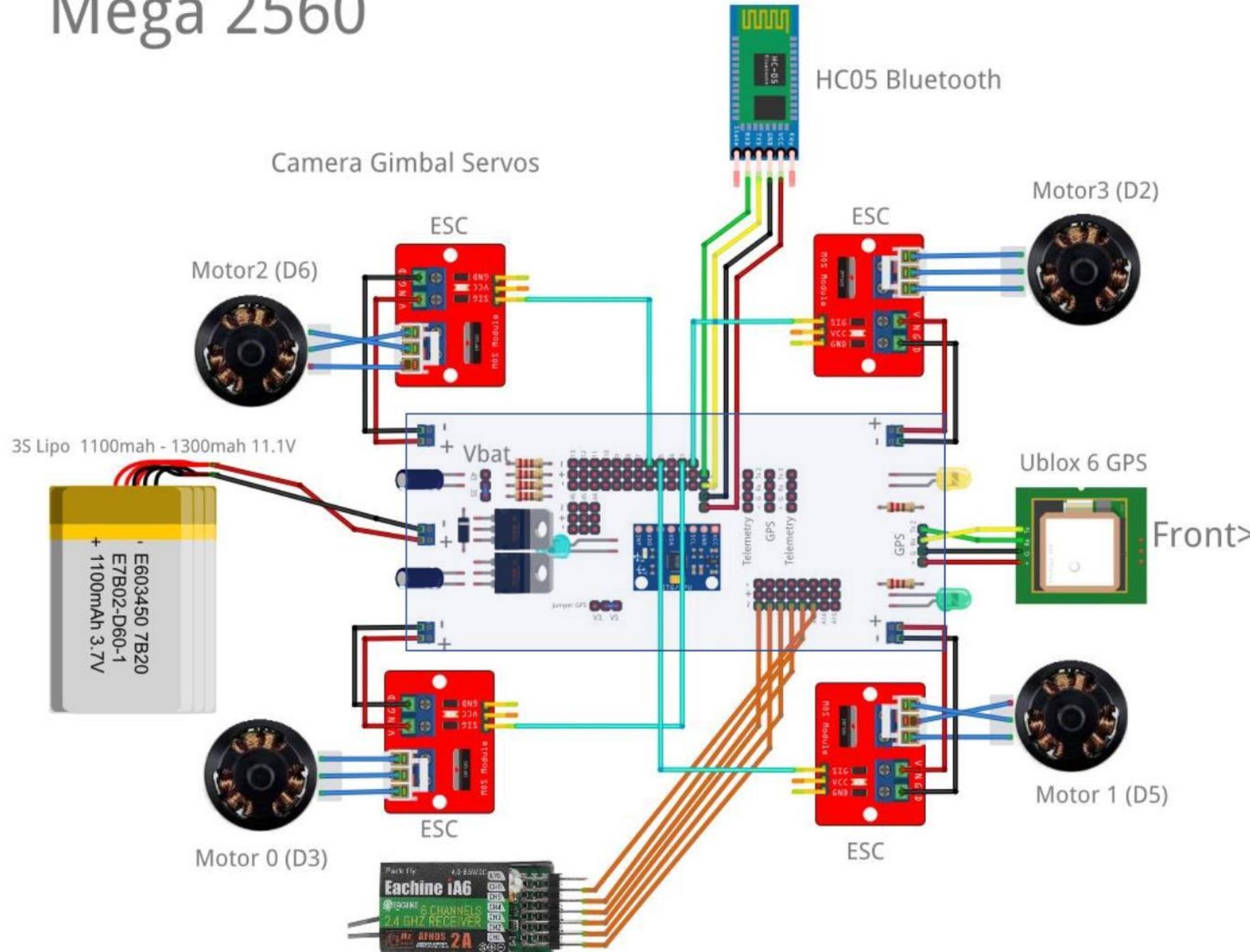
Silver for counter clockwise Motor



UNO 328



Mega 2560



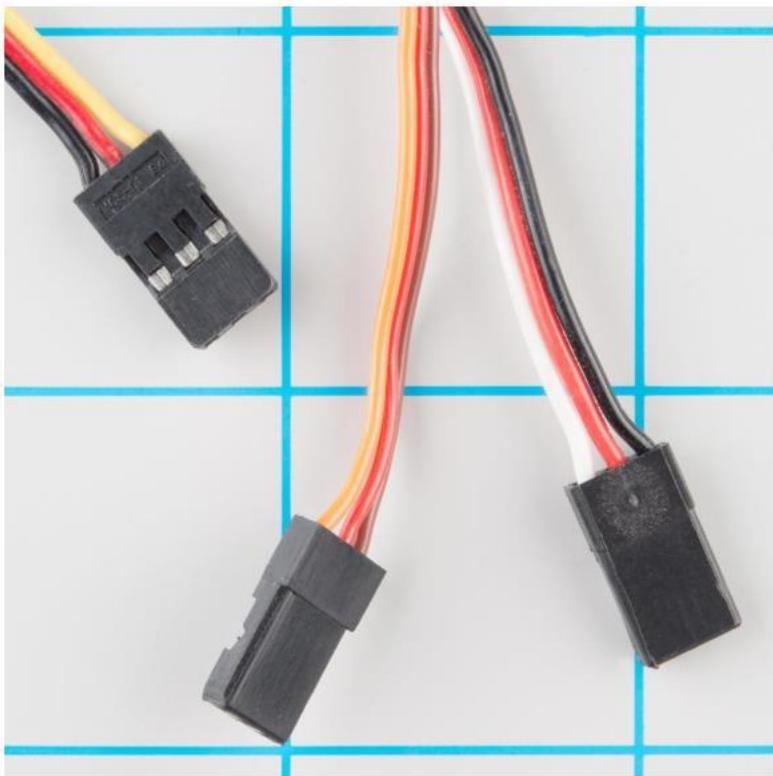
PWM INPUT Assignment

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

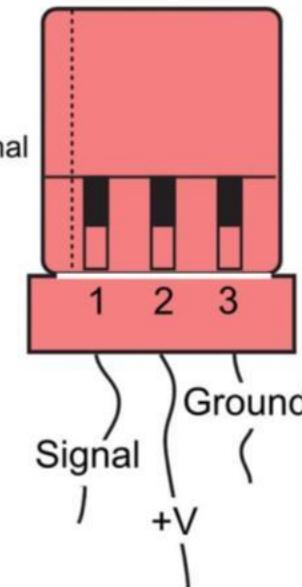


J-type (Futaba)



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



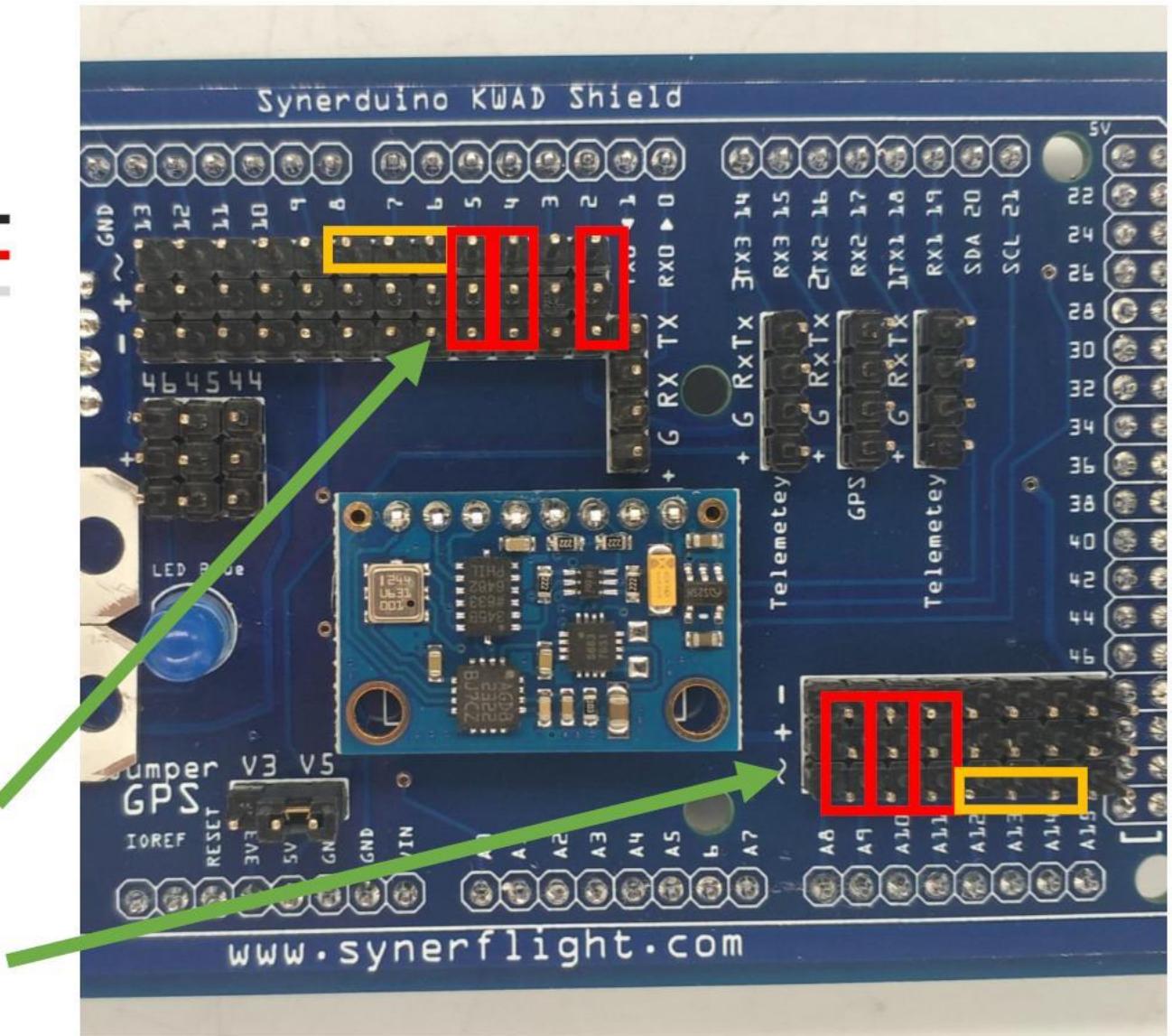
PWM RECEIVER



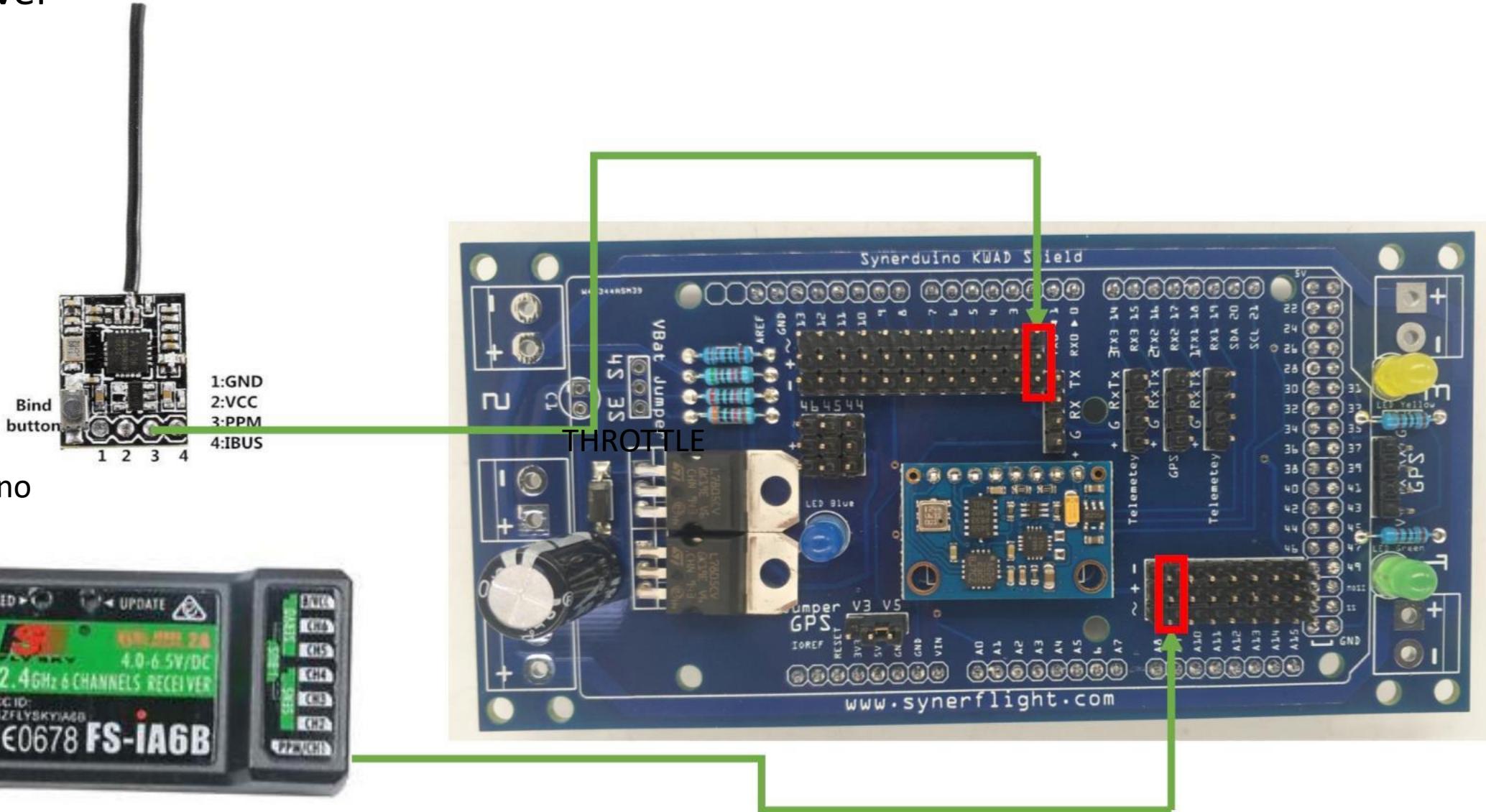
MOSTLY RUDDER AUX1 AND AUX2

UNO PWM IN

MEGA PWM IN



PPM Receiver



Pin D2 for Uno



Pin A8 for Mega

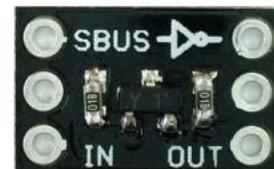
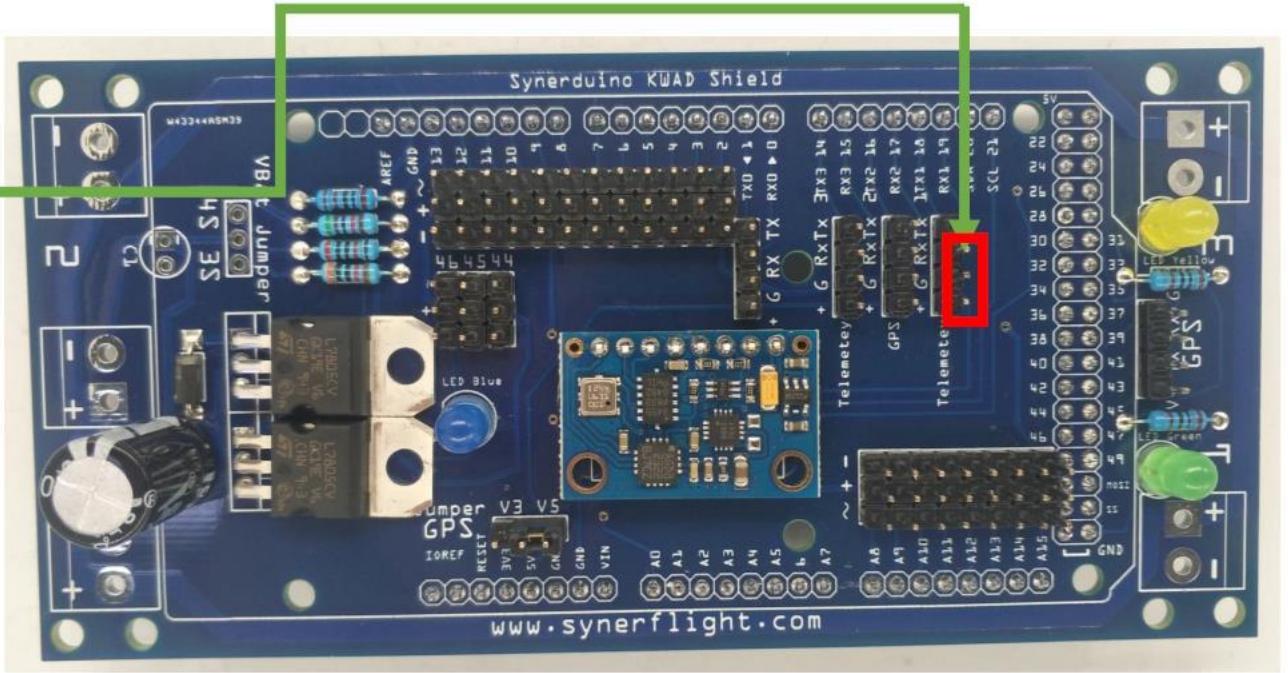
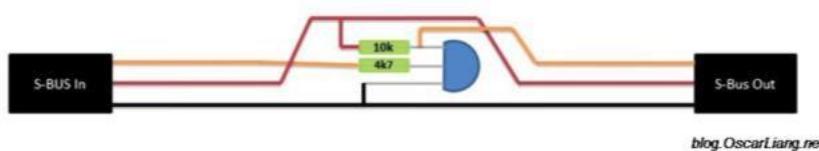
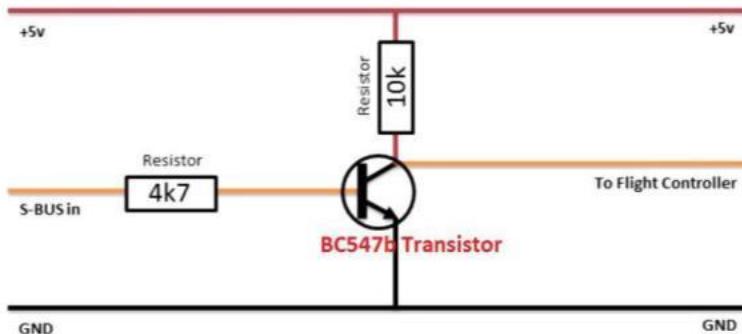
SBUS Receiver

Most modern Receivers now comes with Serial Protocol as they are faster than the old PWM or PPM standard and its now the Modern defacto for Receiver to Flight Control Board communication



RX 1 Telemetry

The SBUS system uses Futaba protocol and should be compatible with Most SBUS Receivers



SBUS Inverter

Should there be issues in Signal Inversion an SBUS inverter may be used

You can tell if the RC control is not being read

SBUS Receiver Inversion

We all get confused sometimes we plug the receiver or PPM/PWM/SBUS Converter in and it suppose to work but it doesn't

SBUS inversion depending on the Brand of Receiver or the PPM/PWM/SBUS Converter you have the SBUS signal can come as Forward Signal or Reverse Signal . This is crucial in getting a Good Receiver connection to the Synerduino STM

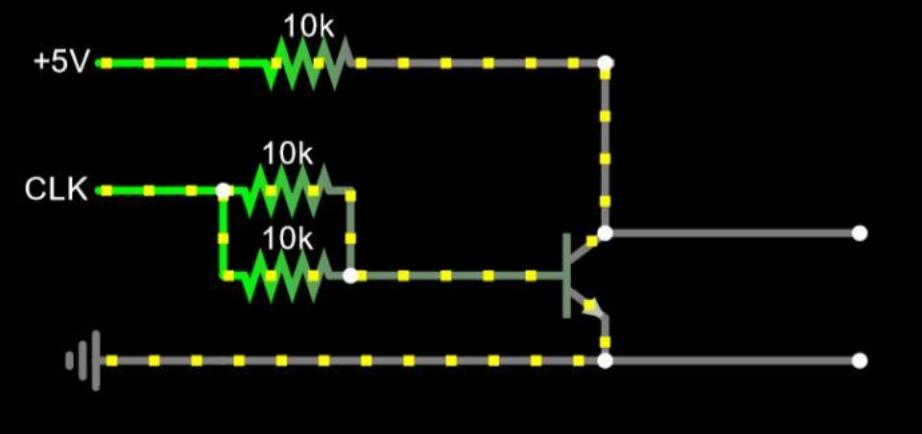
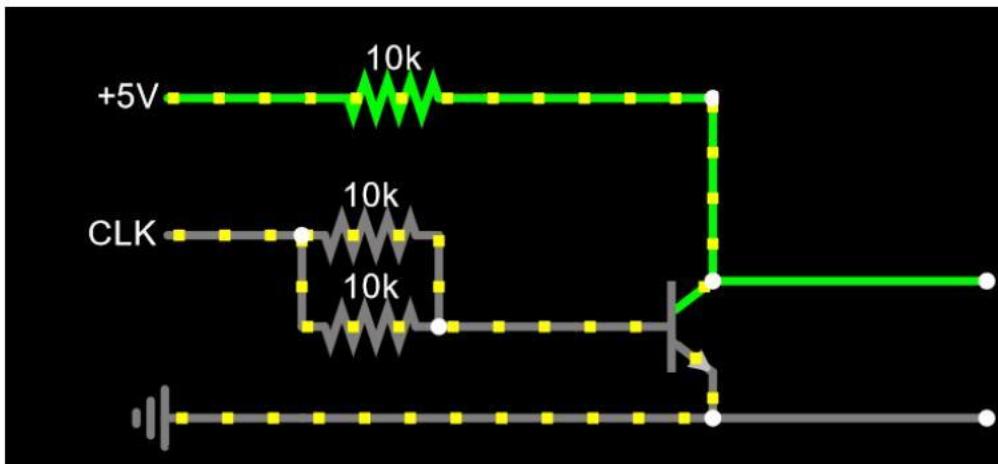
Normal SBUS



Inverted SBUS



Y U No Read SBUS



TELEMETRY RADIO



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



38400 FOR XBEE RADIO

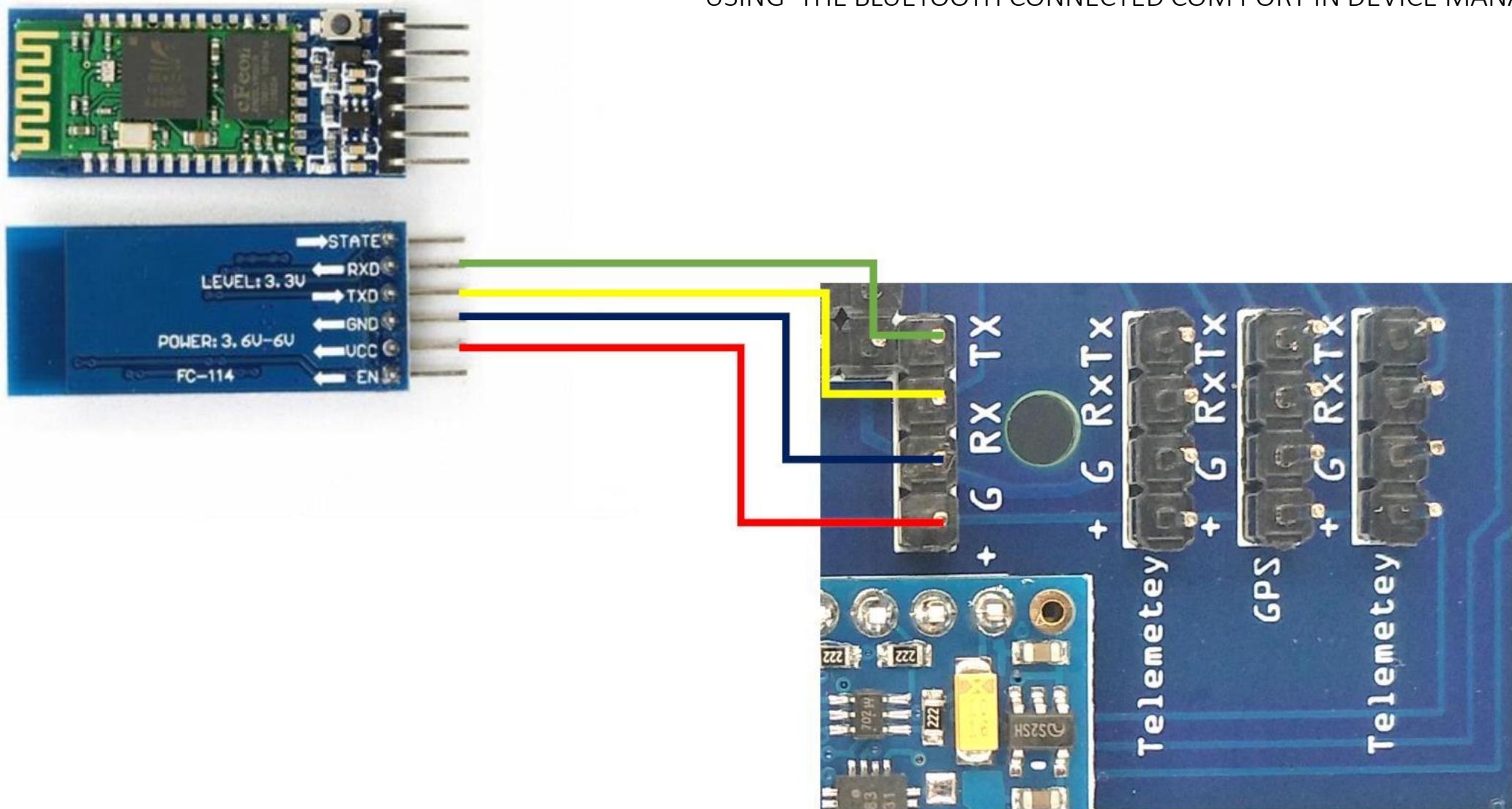


115200 FOR BLUETOOTH HC-05

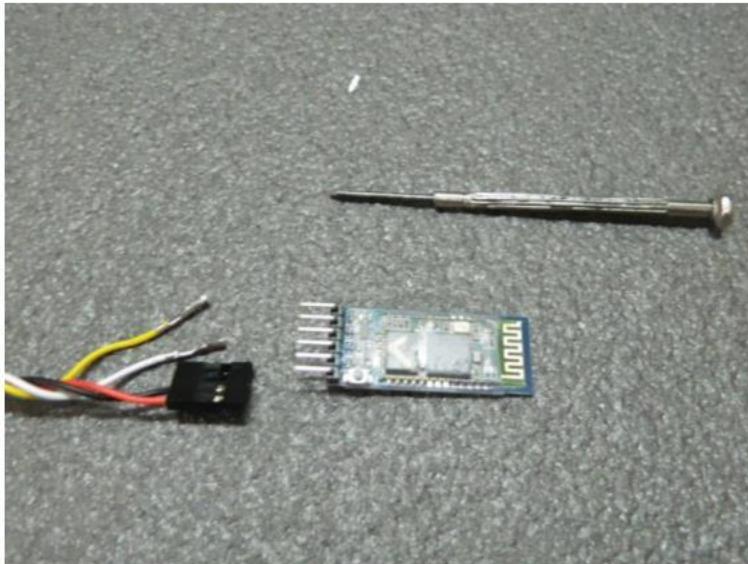
STANDARD FOR ALL DRONES TO USE SERIAL LINK AS TELEMETRY MAINLY ON YOUR TX RX SERIAL PORTS , NOTE: THE LOWER THE FREQUENCY OF THE RADIO THE LOWER THE BAUD IS NEEDED
PROTOCOL IS MSP RAW OR MAVLINK

Bluetooth

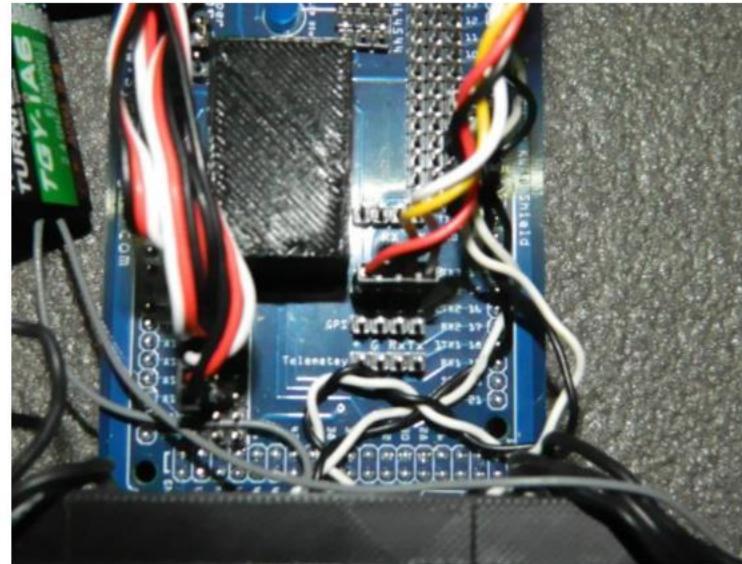
NOTE: USING 0 AT BAUD115200 APPLICABLE BOTH UNO AND MEGA BOARDS REQUIRES THE BLUETOOTH ONLY TO BE PLUG IN AFTER THE SKETCH/FIRMWARE HAS BE UPLOADED TO THE BOARD . AND PAIRING USING THE BLUETOOTH CONNECTED COM PORT IN DEVICE MANAGER



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

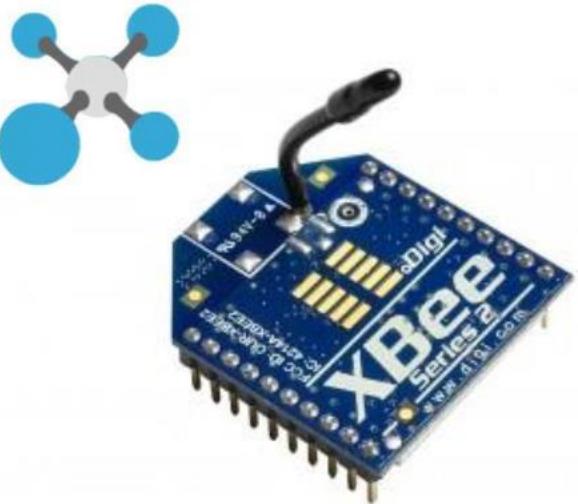


SEE TO IT THE WIRES COLOR CODE MATCHES THE MARKINGS

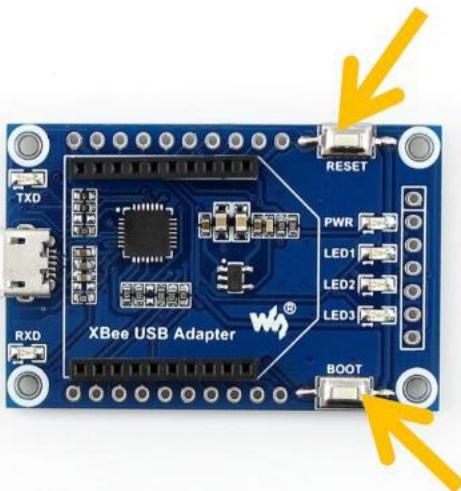
IMPROPER INSTALLATION MAY CAUSE DAMAGE TO THE ARDUINO BOARD AND SHIELD DUE TO REVERSE POLARITY

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

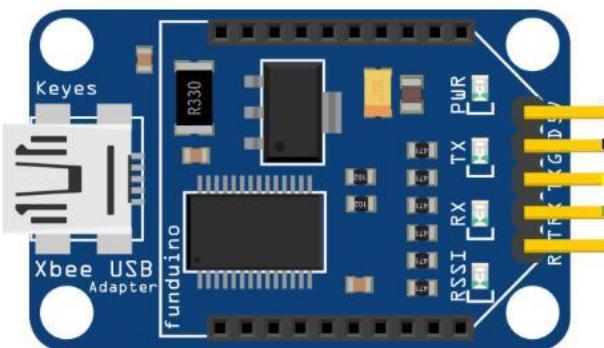
XBEE RADIO



38400 FOR XBEE RADIO



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

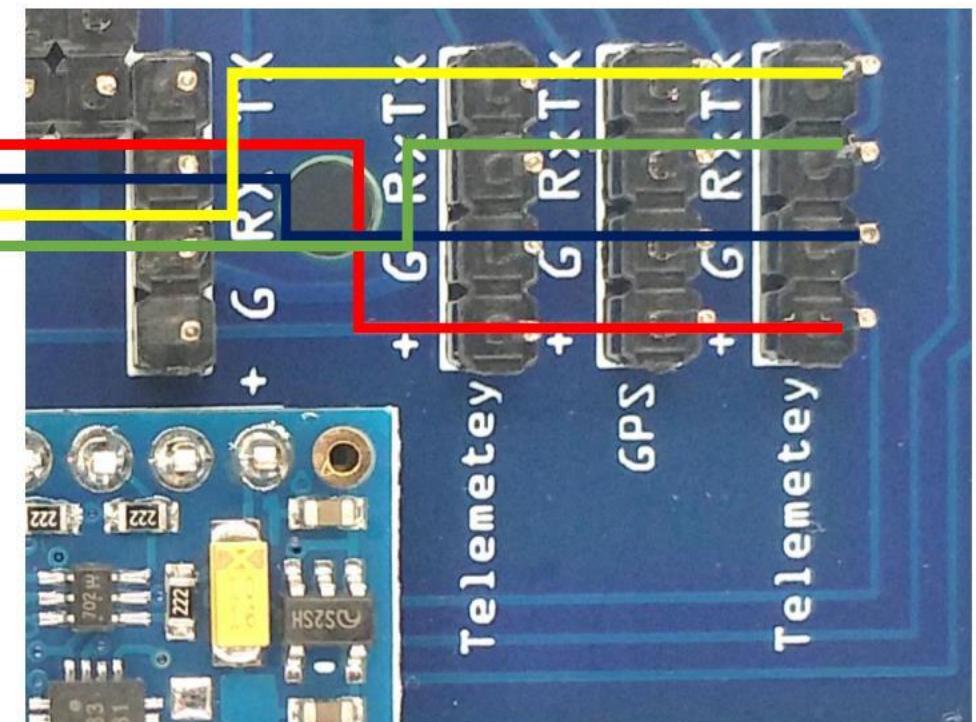


VCC >> +

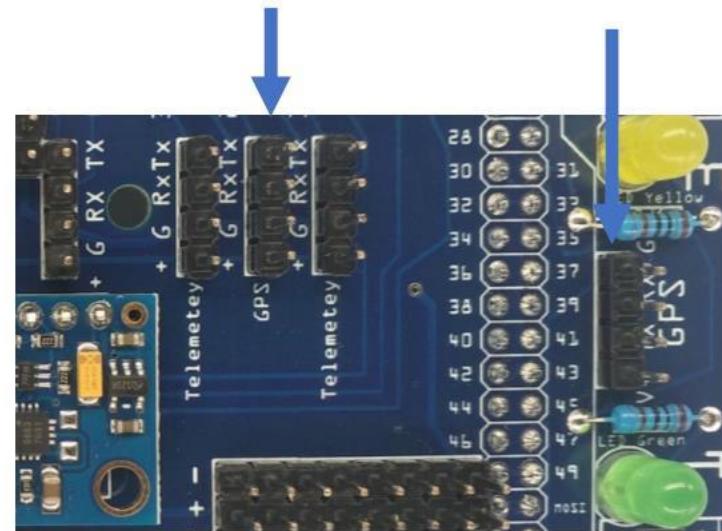
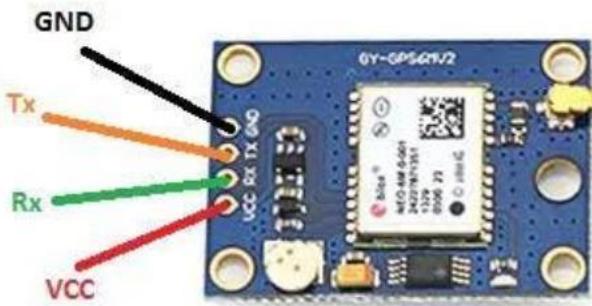
GND >> G

TX >> TX

RX >> RX



GPS



U BLOX NEO 6

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

ATTENTION:

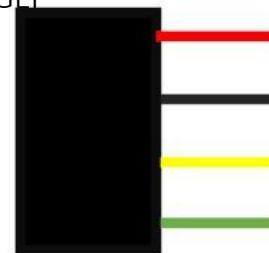
YOU MAY NEED TO REARRANGE THE HEADERS TO CONNECT THE GPS MODULE TO THE SHIELD BOARD ACCORDINGLY

VCC >> +

GND >> G

TX >> RX

RX >> TX



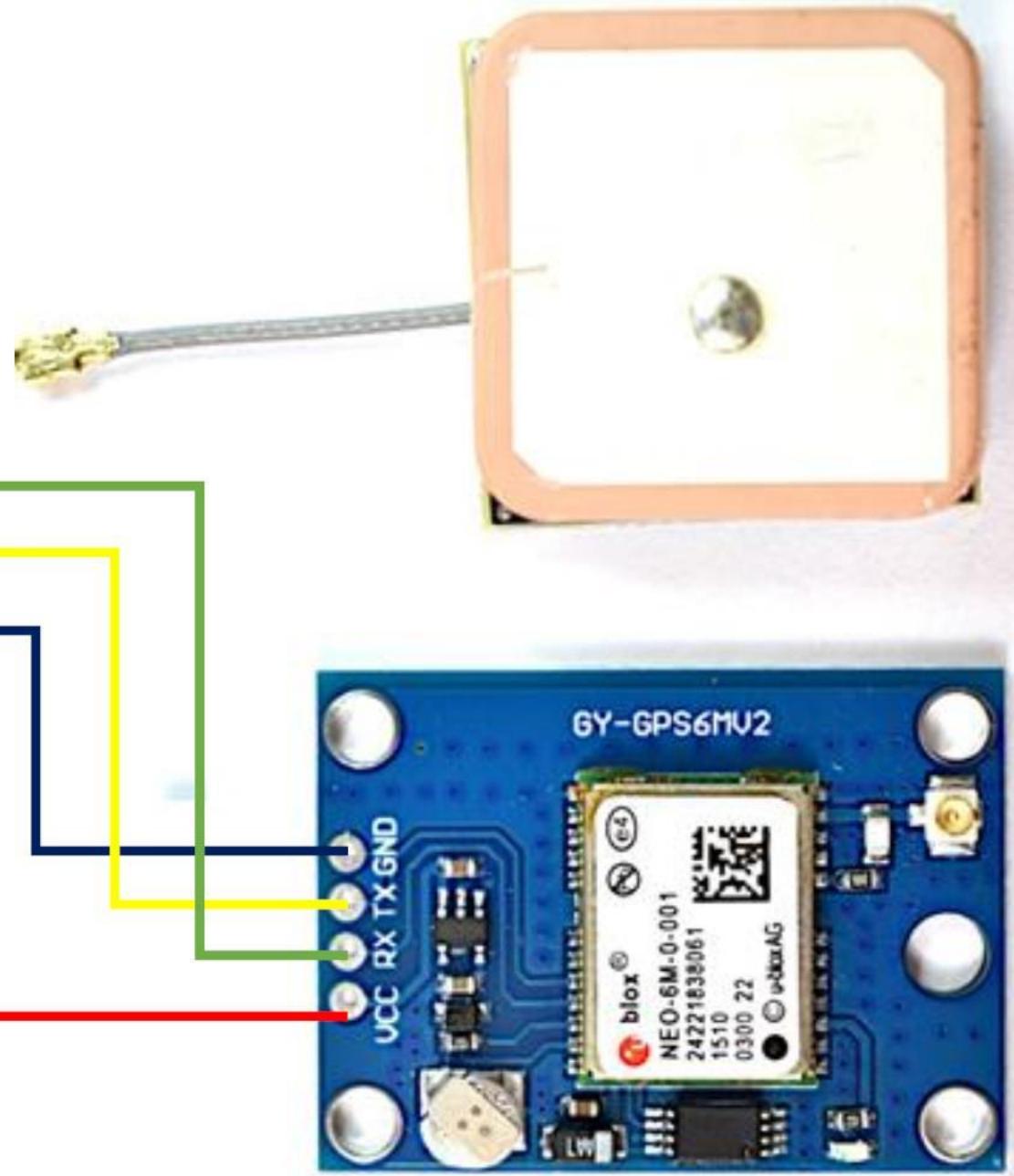
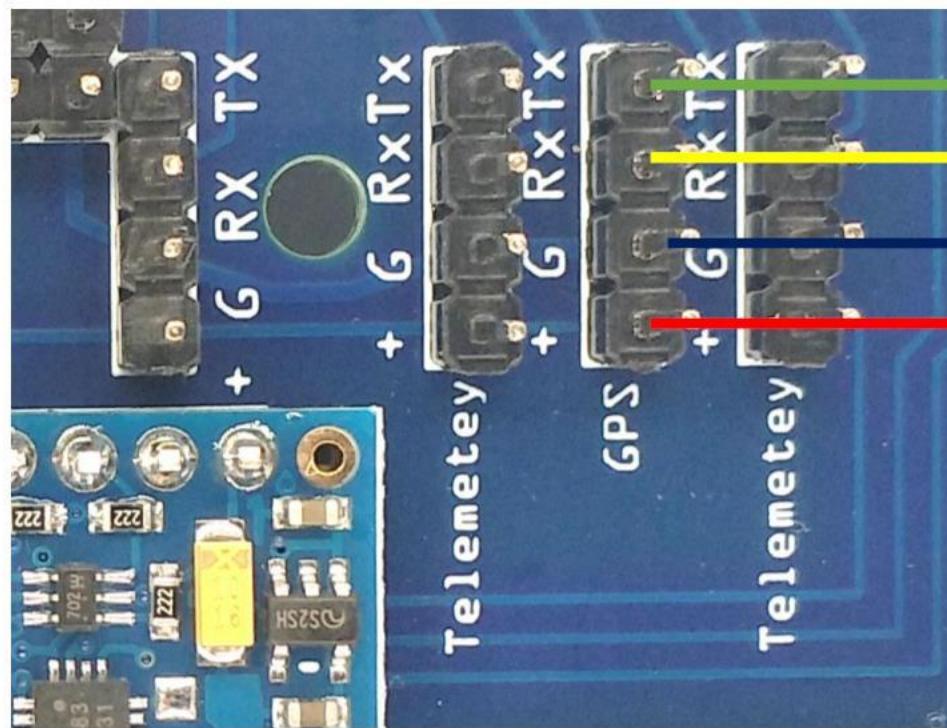
SEE TO IT THE WIRES COLOR CODE MATCHES THE MARKINGS

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

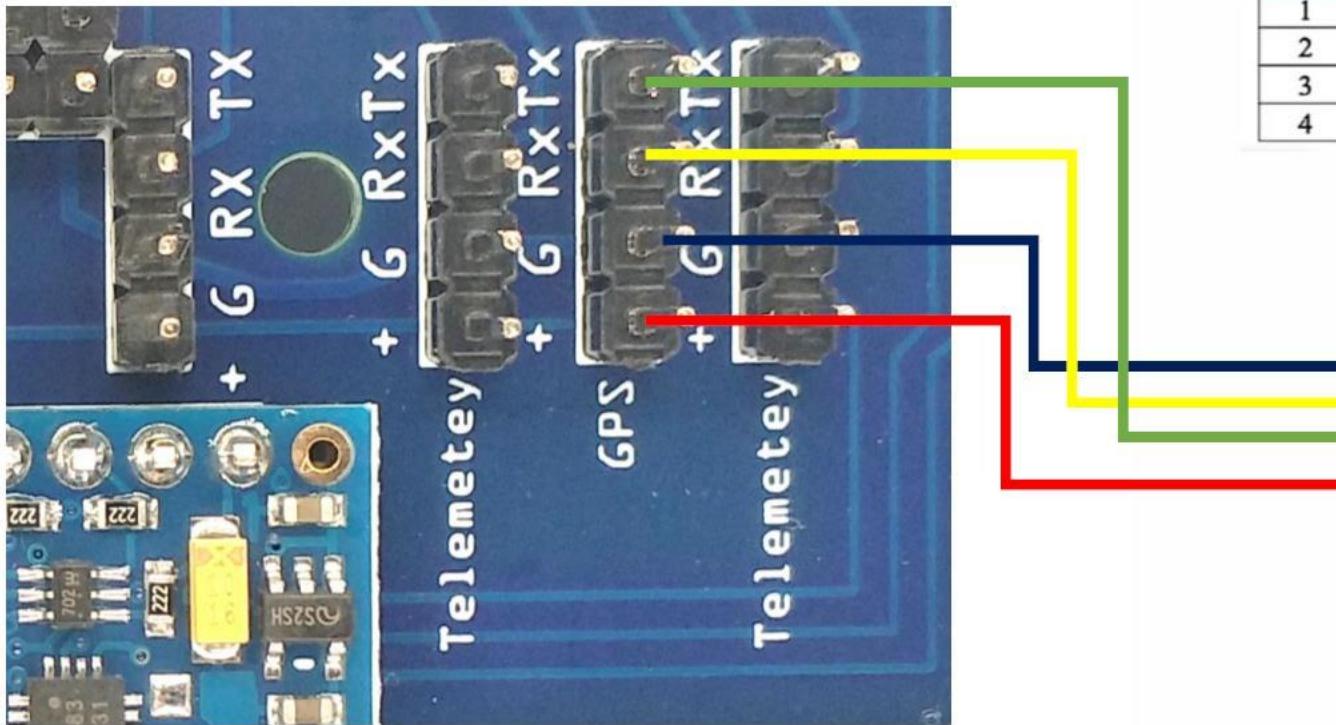
GPS NEO6 (NMEA)



GPS BEITIAN (UBLOX)



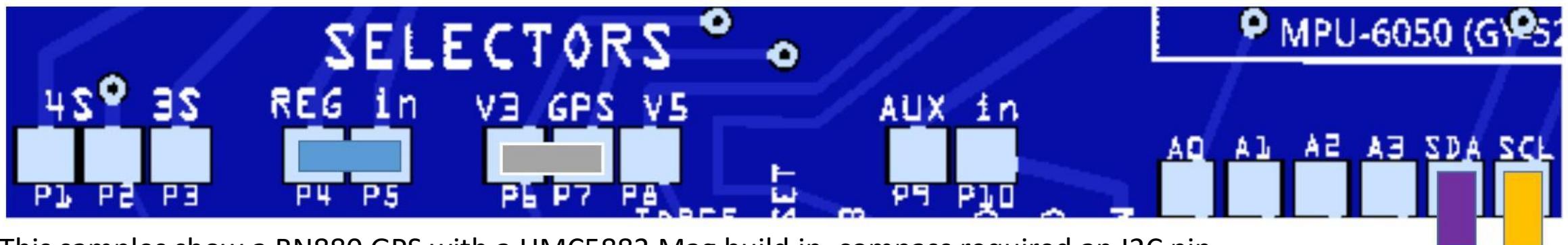
4.VCC
3.RX
2.TX
1.GND



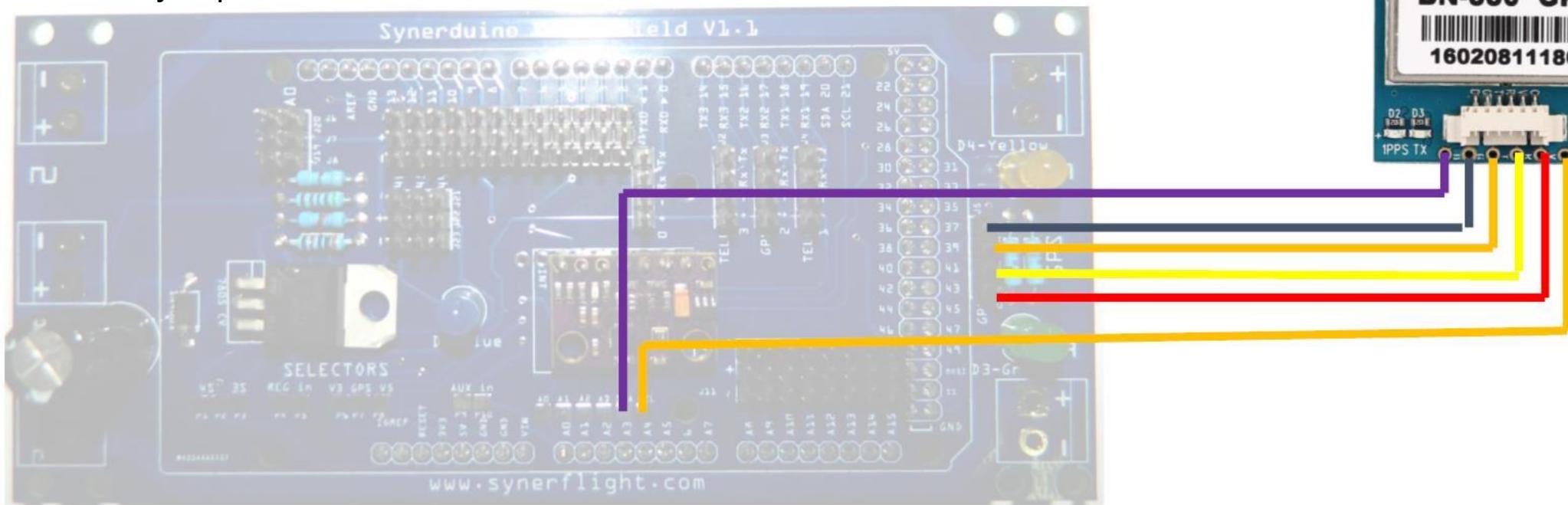
PIN	PIN Name	I/O	Description
1	GND	G	Ground
2	TX	O	Serial Data Output.
3	RX	I	Serial Data Input.
4	VCC	I	DC 3.0V - 5.5V supply input,Typical: 5.0V



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



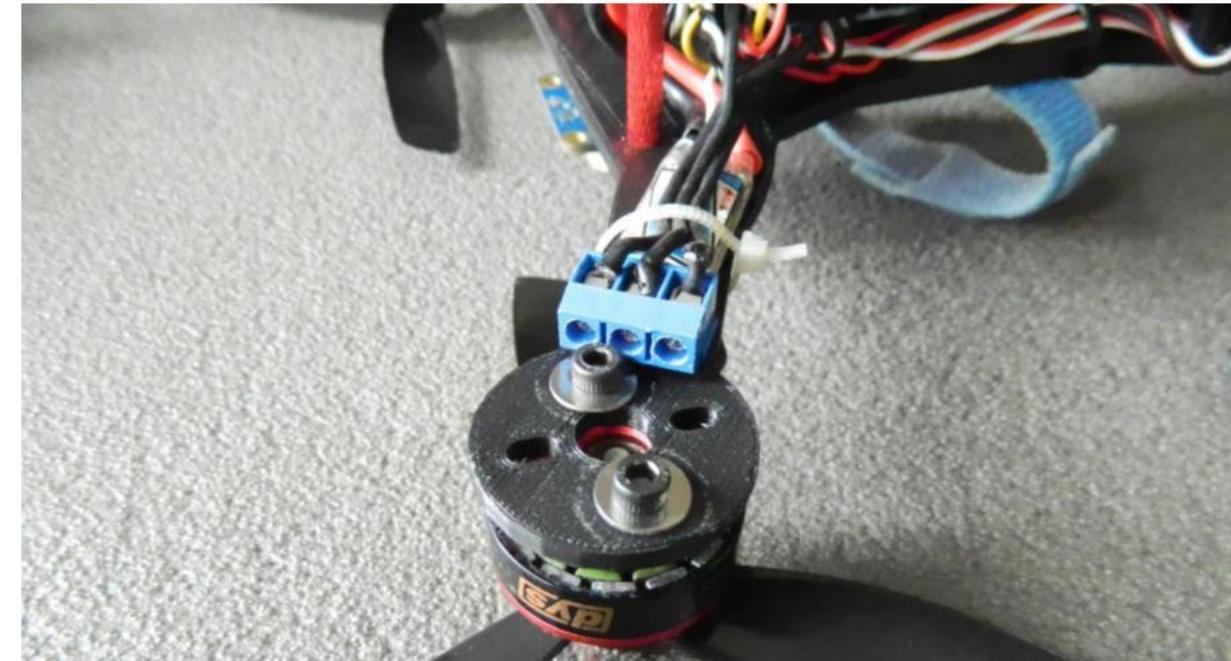
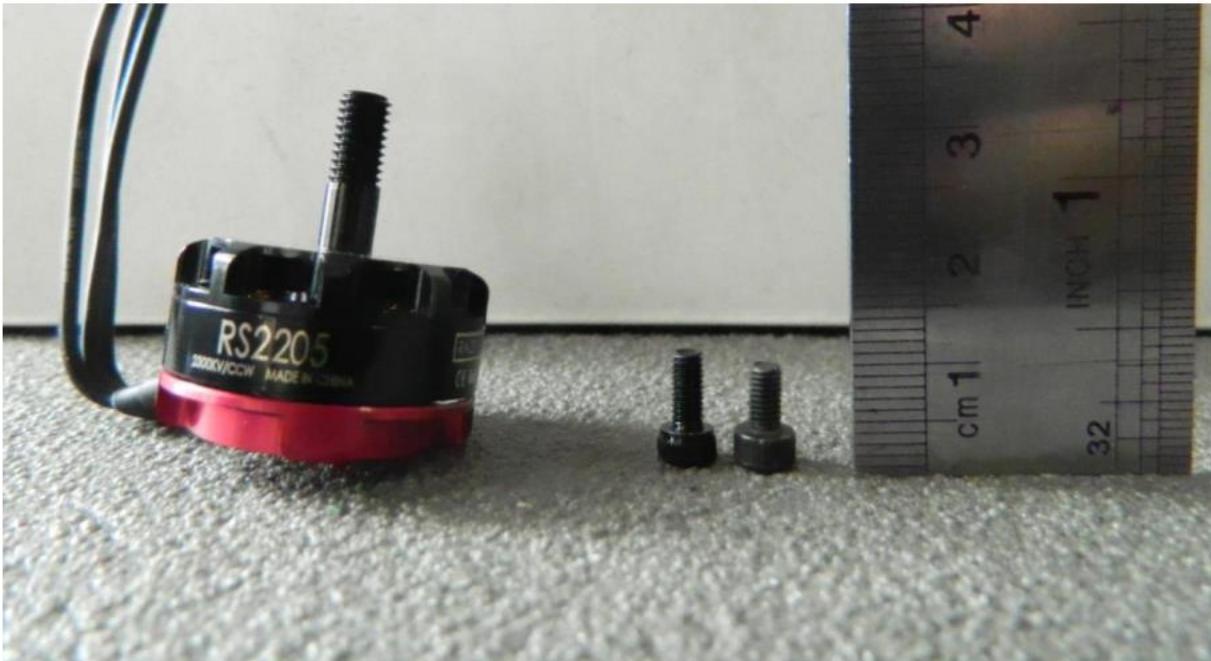
Arduino Board Preparation motor installation



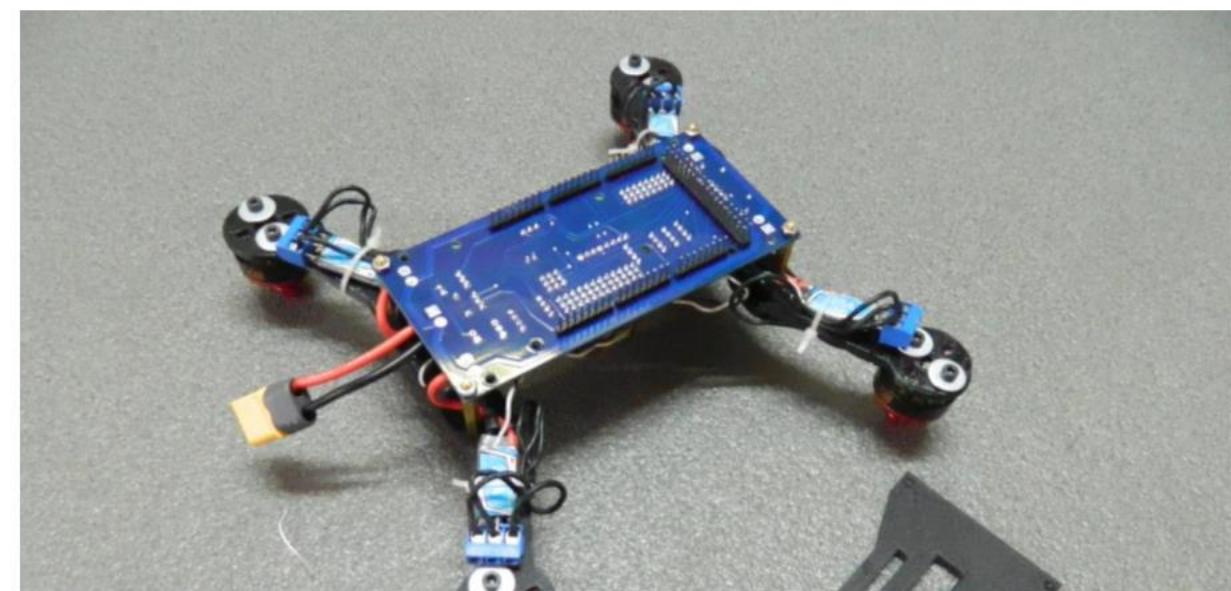
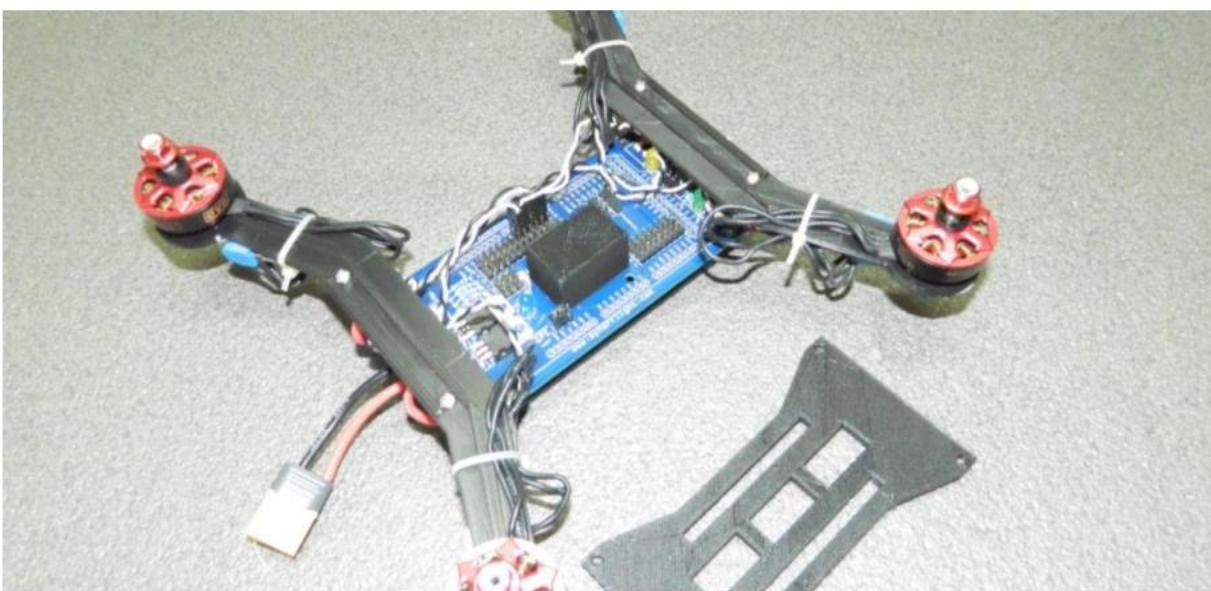
2560 MEGA

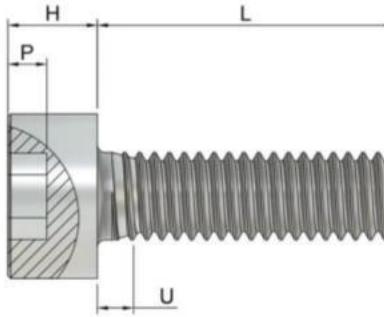
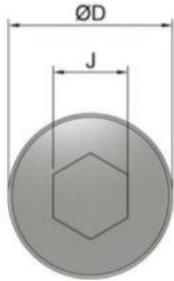


UNO 328



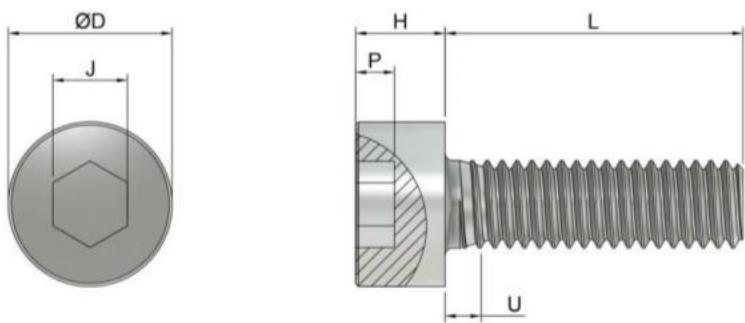
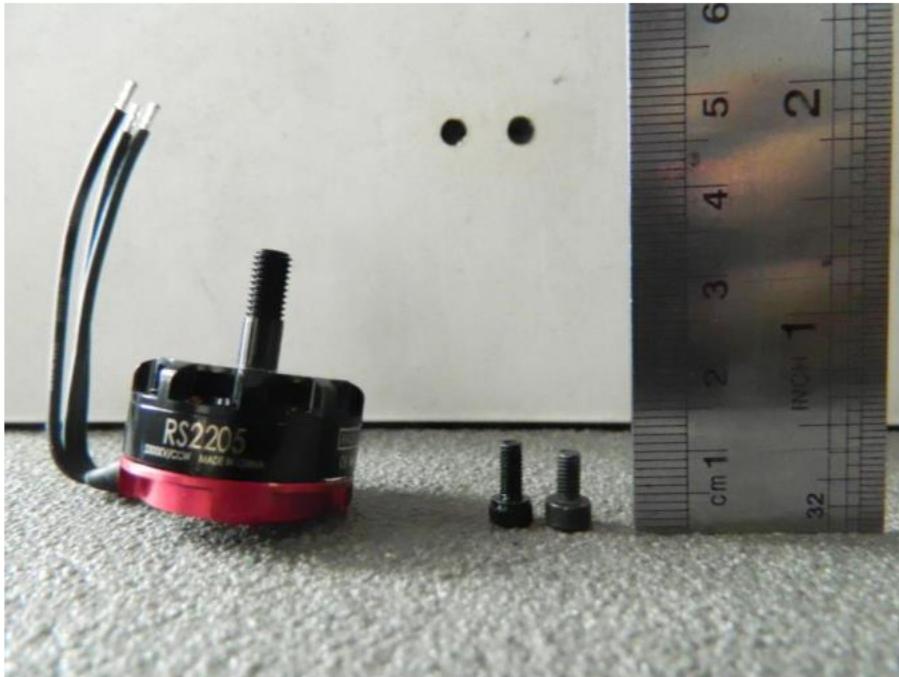
Use the 6mm M3 Bolt provided (short one) and install with washers





Thread Lock Purple or PVA Glue

Vibrations can cause bolts/Screws to come loose especially when bolts are on smooth plastic surface
Low strength thread lock or PVA glue are suitable in holding the bolts in place yet temporary enough that can still be removed by hand tools. Secure motors , PCB board and frame by applying a few drops enough to coat the thread of the screw and slowly tighten it in



Note : this frame is design to use M3 bolts with thread length of 6mm with washer as fittings included in the set (use the shortest bolt that came with the motor set)

Ensure that the bolt thread does not touch the inner wire coils of the motor

Use small amount of PVA white Glue to Thread Lock the bolt in place Preventing it from going loose

How to Choose Drone Motors



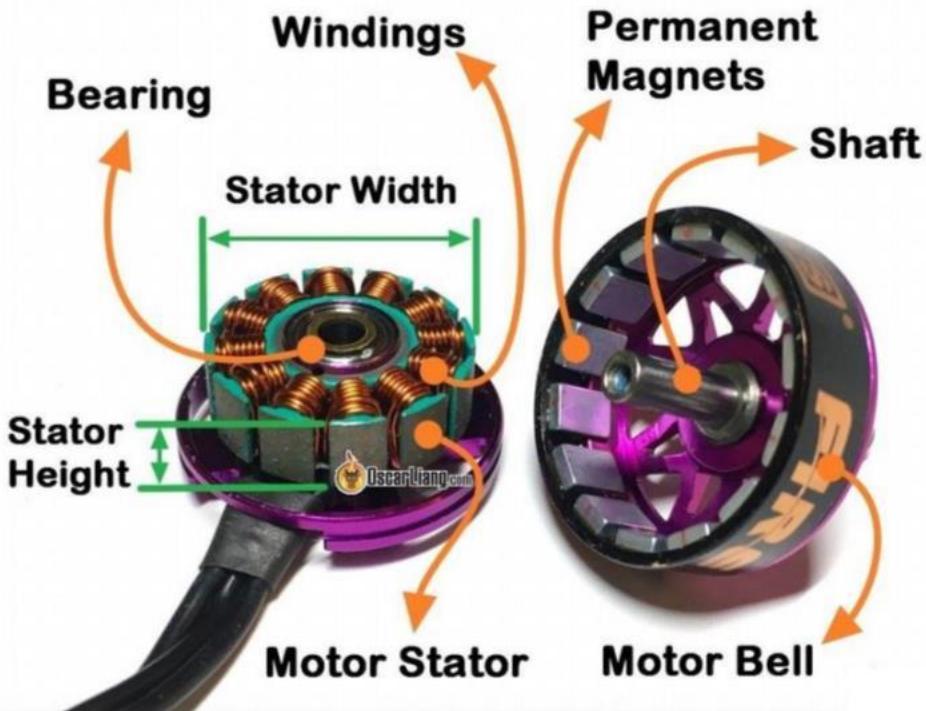
Before choosing motors, you should at least have a rough idea of the size and weight of the drone you want to build. I will explain the process of determining motor size based on drone size, Here is a rough chart of what to expect for a given frame size

Frame Size	Prop Size	Motor Size	KV
150mm or smaller	3" or smaller	1105 -1306 or smaller	3000KV and higher
180mm	4"	1806, 2204	2600KV – 3000KV
210mm	5"	2205-2208, 2305-2306	2300KV-2600KV
250mm	6"	2206-2208, 2306	2000KV-2300KV
350mm	7"	2506-2508	1200KV-1600KV
450mm	8", 9", 10" or larger	26XX and larger	1200KV and lower

Motor Size Explained

The size of brushless motors in RC is normally indicated by a 4-digit number – AABB. “AA” is the **stator width** (or stator diameter) while “BB” is the **stator height**, both are measured in millimeter.

KV is RPM Per Volt in this sample its 2300kv



Brand Name Rotation Arrow (CW)



2300KV/CW BR2205 Motor Model Number Stator Width Stator Height

Motors Current and Voltage

It's important to understand that voltage has a large impact on your motor and propeller choice. Your motor will try to spin faster with a higher voltage, thus draw a higher current. Ensure you are aware of how much thrust your motors produce and how much current they will draw.



Quadcopter flight time = (Battery Capacity * Battery Discharge /Average Amp Draw)*60

**1300Mah 11.1V Lipo
80% Discharge**

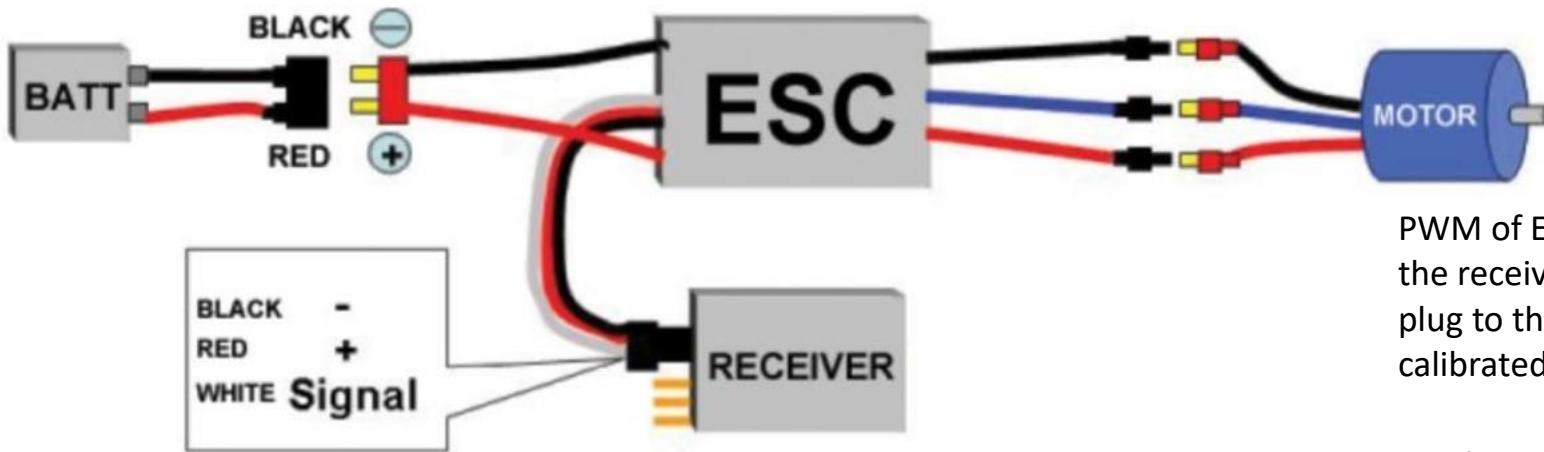
**Sample base on motor datasheet max rating :
Round off 5A per motor
 $(1300\text{mah}/1000\text{mah}) * 80\% / 20\text{Amps} * 60 \text{ Min}$
= 3.12min (Note motor chars specify Max amp)**

MOTOR TECHNICAL DATA:

VOLTAGE V	NO LOAD		ON LOAD				LOAD TYPE Battery/prop
	CURRENT A	SPEED rpm	CURRENT A	Pull g	Power W	EPP %	
12	1.2	30000	5.3	200	63.6	3.1	LiPo3/4045x3
			13.3	400	159.6	2.5	
			22.7	580	272.4	2.1	
			5.7	250	68.4	3.7	LiPo3/5045x4
			14.2	500	170.4	2.9	
			31.6	800	379.2	2.1	
12	1.2	30000	5.1	250	61.2	4.1	LiPo3/5040x3
			12.2	500	146.4	3.4	
			21.6	720	259.2	2.8	
			5.2	250	62.4	4.0	LiPo3/5040x3
			12.9	500	154.8	3.2	
			23.8	750	285.6	2.6	
19.6	700	235.2	4.9	250	58.8	4.3	LiPo3/5045
			11.9	500	142.8	3.5	
			19.6	700	235.2	3.0	

Note : most motor technical Data would only present maximum current draw rating of the motor and given your not always on full throttle all the time in flying so we can give the current draw a little less than what is shown here

Electronic Speed Controller CALIBRATION



Propellers are removed during this process

PWM of ESC is directly hook up to the receiver Throttle pin . Ensure the receiver is getting power thru the Aux PWM pin which remains plug to the synerduino board (process is repeated till all ESCs are calibrated)

Multirotors must have all ESCs calibrated similarly to ensure reliable operations

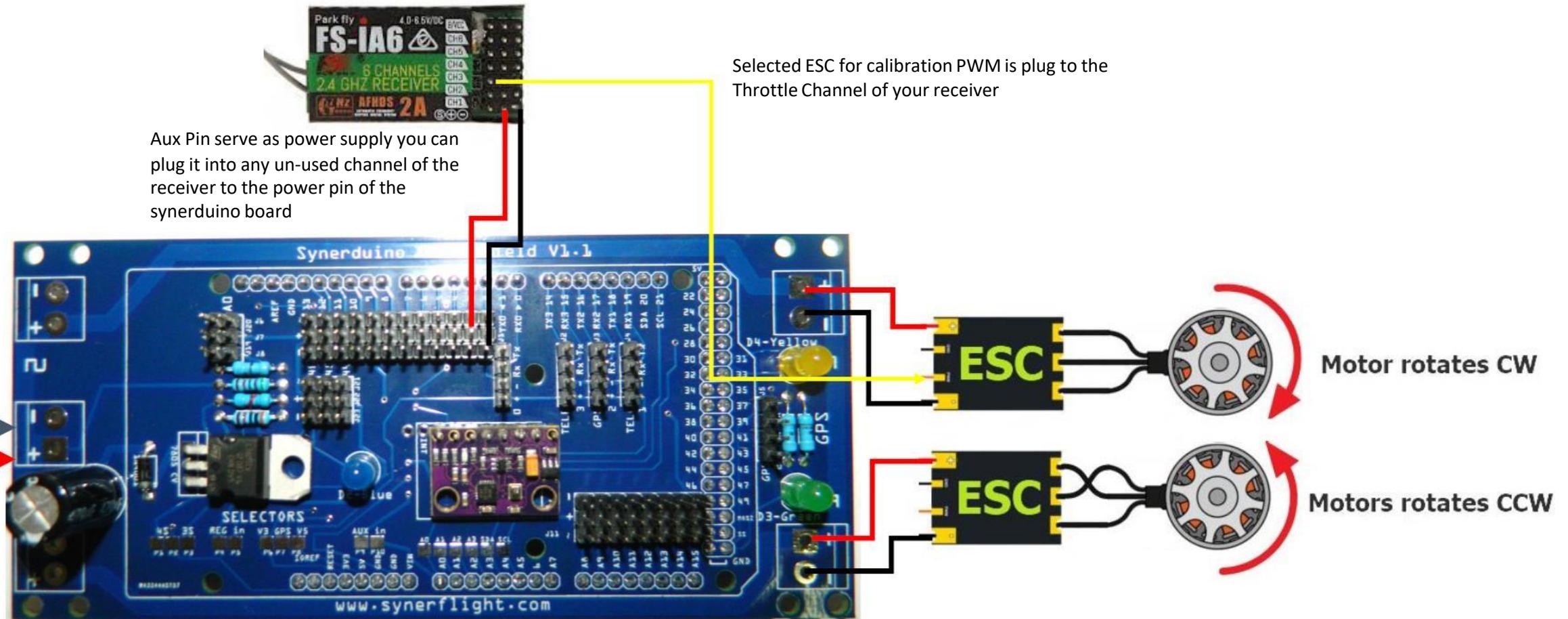
Motor must be plug in at this point w/o the propeller. As it will serve several purpose

- An Speaker to listen to calibration tone of the ESC
- Identify motor rotation should it needs to be corrected
- Test full speed range

ESC calibration will vary based on what brand of ESC you are using, so always refer to the documentation for the brand of ESC you are using for specific information (such as tones). “All at once” calibration works well for most ESCs, so it is good idea to attempt it first and if that fails try the “Manual ESC-by-ESC” method.

If your ESC happens to be an OPTO. The Synerduino board can provide as power supply for both RC Receiver and ESCs when soldered in . Get the PWM Pin of the ESC you want to calibrate and plug it into the Throttle Channel of your Receiver

Individual ESC calibration method , for individual ESC at a time



Electronic Speed Controller CALIBRATION

1.



Turn transmitter on.
Set throttle to maximum.

You have the option to put
your trim center or lowest
trim setting using the trim
buttons

2.



Connect battery to power module.
/ Synerduino Shield board

3.



Wait for your ESCs/Motor to emit the musical tone, the regular number of beeps indicating your battery's cell count (i.e. 3 for 3S, 4 for 4S) and then an additional two beeps to indicate that the maximum throttle has been captured.

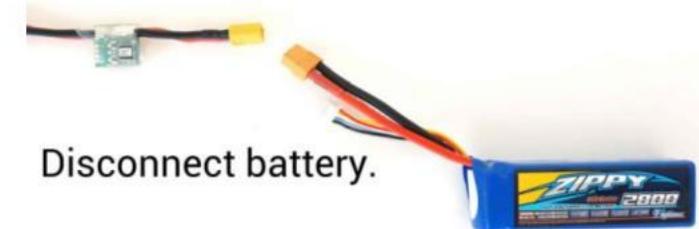
4.



Set throttle to minimum.

Some ESC will give a tone to identify the calibration mode is exited

6.



Disconnect battery.

5.



7.



Connect battery to power module.

Reconnect the battery with the throttle to minimum to initiated booth up then test the throttle. This Process is repeated till all ESC are calibrated

Synerduino Multiwii ESC calibration method , for All ESC at Once

Throttle Pin is Connected to the Throttle Channel of the Receiver

All other Channels as is



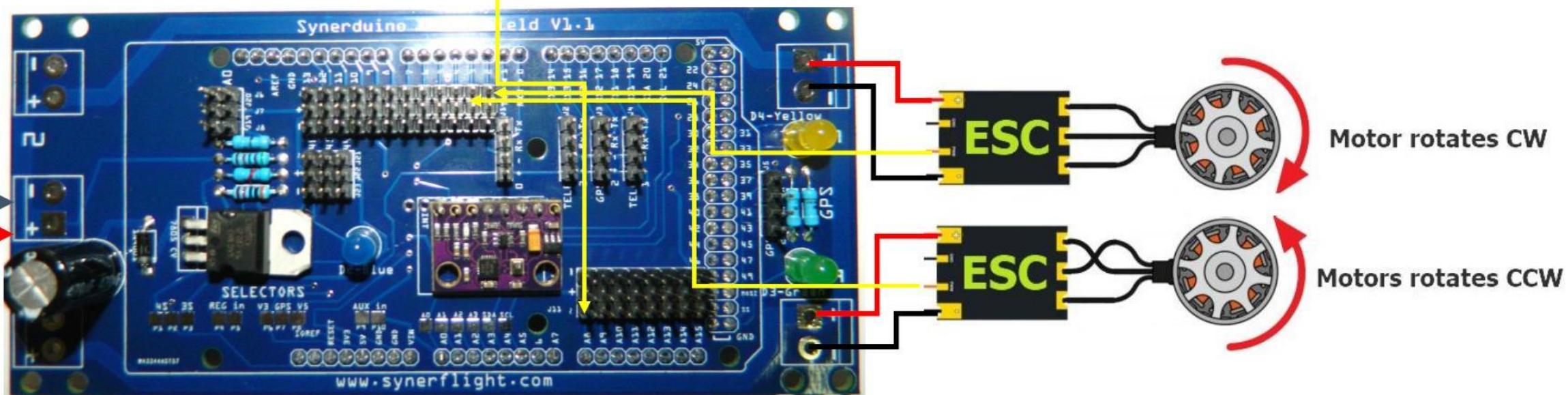
to calibrate all ESCs connected to MWii at the same time (useful to avoid unplugging/re-plugging each ESC)

Warning: this creates a special version of MultiWii Code

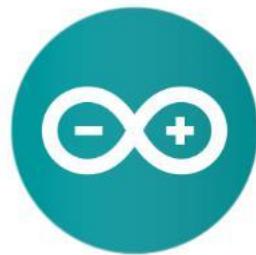
You cannot fly with this special version. It is only to be used for calibrating ESCs

This is applicable to those who have PPM and SBUS Receivers

Plug to battery 3S



MULTIWII.INO



Multiwii Sketch

source code for Synerduino KWAD Shield: compile this using the arduino IDE then flash it to your Arduino board

(Arduino 1.8.16) SynerduinoKwad4-GY801-GY91-1.8.16 [Download](#)

()Arduino 1.8.16 – 2.0.0 SynerduinoGY91-1.8.6-2.0.0 [Download](#)

(For UNO boards)Synerduino-Uno-GY91-801 [Download](#)

<http://synerflight.com/kwad-documentation/3-software/>

The Calibration sketch is accessible on these firmware config.h Tab

This houses the bulk of that the Arduino Drones codes

CONFIG.H



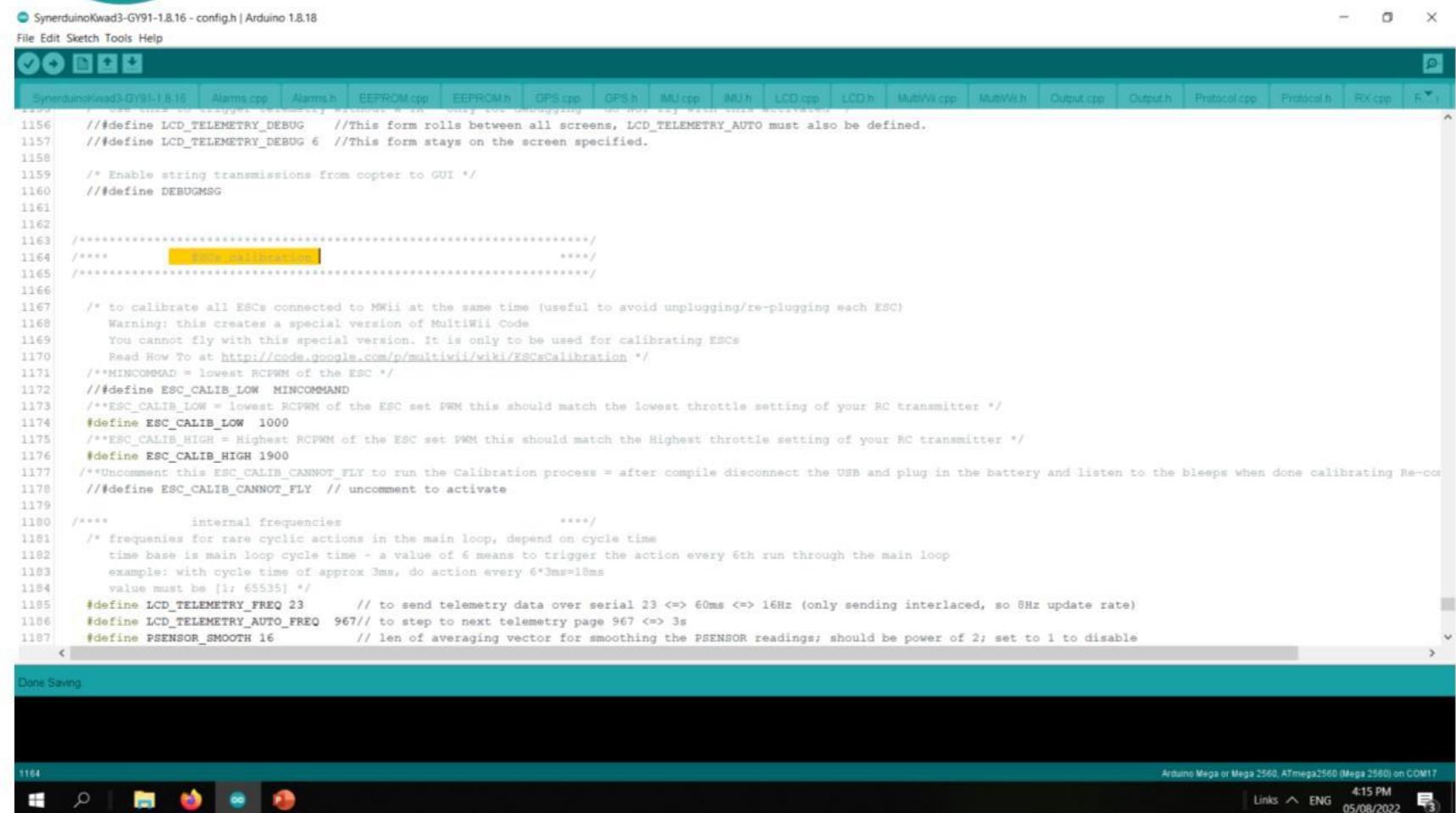
In Config.h Tab

Look for ESC Calibration Menu

Uncomment Define ESC_Calib_Cannot_Fly

This disable all other functions of the drone other than the Throttle input from the receiver as Pass thru.

Remember to Re-comment this menu after Calibration Process is complete



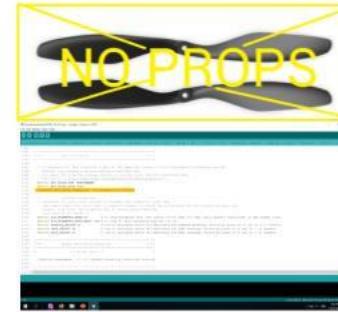
```
1156 // #define LCD_TELEMETRY_DEBUG //This form rolls between all screens, LCD_TELEMETRY_AUTO must also be defined.
1157 // #define LCD_TELEMETRY_DEBUG 6 //This form stays on the screen specified.
1158
1159 /* Enable string transmissions from copter to GUI */
1160 // #define DEBUGMSG
1161
1162 ****
1163 /* ESC Calibration */
1164 ****
1165 ****
1166
1167 /* to calibrate all ESCs connected to Mwii at the same time (useful to avoid unplugging/re-plugging each ESC)
   Warning: this creates a special version of Multiwii Code
   You cannot fly with this special version. It is only to be used for calibrating ESCs
   Read How To at http://code.google.com/p/multiwii/wiki/ESCalibration */
1168 /**MINCOMMAND = lowest RCPWM of the ESC */
1169 // #define ESC_CALIB_LOW MINCOMMAND
1170 /**ESC_CALIB_LOW = lowest RCPWM of the ESC set PWM this should match the lowest throttle setting of your RC transmitter */
1171 #define ESC_CALIB_LOW 1000
1172 /**ESC_CALIB_HIGH = Highest RCPWM of the ESC set PWM this should match the Highest throttle setting of your RC transmitter */
1173 #define ESC_CALIB_HIGH 1900
1174 /**Uncomment this ESC_CALIB_CANNOT_FLY to run the Calibration process = after compile disconnect the USB and plug in the battery and listen to the bleeps when done calibrating Re-connect the USB */
1175 // #define ESC_CALIB_CANNOT_FLY // uncomment to activate
1176
1177 **** internal frequencies ****
1178 /* frequenies for rare cyclic actions in the main loop, depend on cycle time
   time base is main loop cycle time - a value of 6 means to trigger the action every 6th run through the main loop
   example: with cycle time of approx 3ms, do action every 6*3ms=18ms
   value must be [1: 65535] */
1179
1180 #define LCD_TELEMETRY_FREQ 23 // to send telemetry data over serial 23 => 60ms => 16Hz (only sending interlaced, so 8Hz update rate)
1181 #define LCD_TELEMETRY_AUTO_FREQ 967// to step to next telemetry page 967 => 3s
1182 #define PSENSOR_SMOOTH 16 // len of averaging vector for smoothing the PSENSOR readings; should be power of 2; set to 1 to disable
1183
1184
1185
1186
1187
```

Done Saving.

Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM17
Links ENG 4:15 PM 05/08/2022

Electronic Speed Controller CALIBRATION

1.remove props or tie copter down.



2.setup all options in config.h to whatever suits your copter

3.activate the *define ESC_CALIB_CANNOT_FLY*, possibly set high and low values for ESC calibration, if you know what you are doing

4.compile, upload, run --- cannot fly and will use Buzzer/LEDs to indicate finished calibration (after approx 10 seconds)

5.comment the define again, compile, upload

6.test carefully with your ESCs calibrated, fly and have fun



Connect battery to power module.



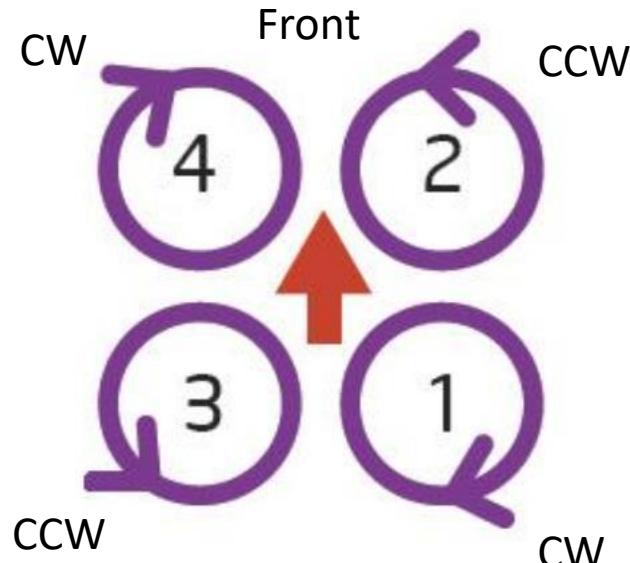
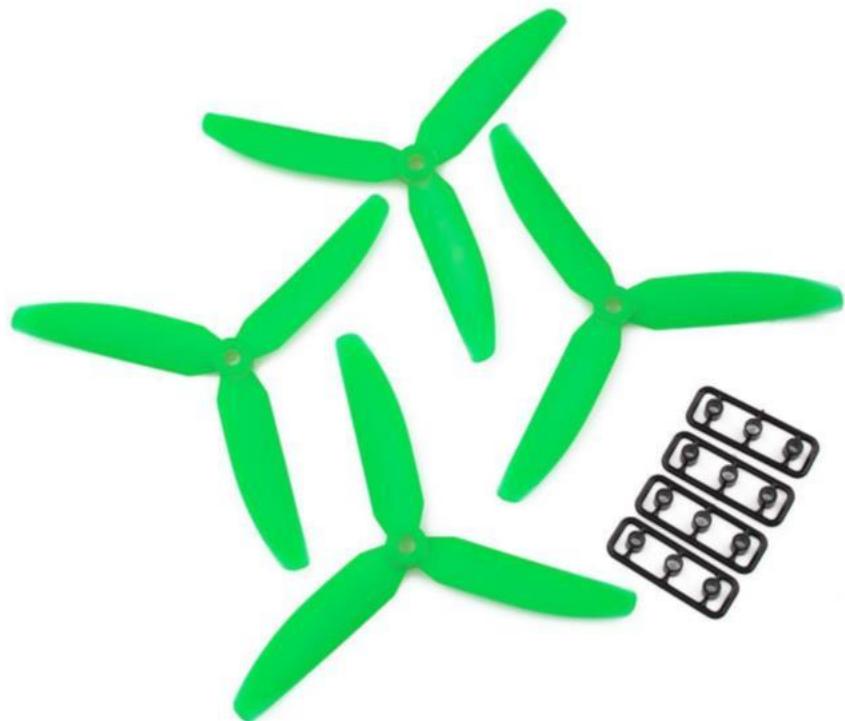
Disconnect battery.



Connect battery to power module.

Propellers

Sometimes even the most prominent features can be overlooked yet this is a critical element on what makes a multirotor fly



QuadCopter-X
(default)

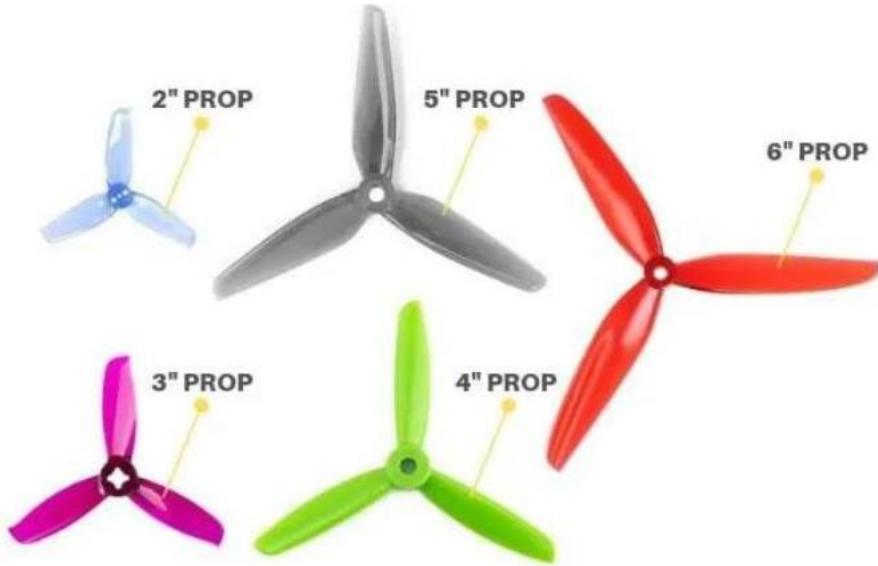
Note: ensure the props are well balanced
With no blade damage if you want to ensure a good Stability in GPS and Altitude hold modes.

Vibration in the frame can cause the sensors to register noise making flights unstable

With the Synerduino 250mm we recommend the 5x45x3 (5045 3) or 5x40x3 (5040 3) Prop type to match with the 2300kv motor the Props have to be stiff too to accommodate the extra load of the quad

Size

Propeller size directly influences thrust and responsiveness.



Smaller propellers are lighter and work against less air. This makes them more responsive; they're easy to stop and speed up.

They require high KV motors to operate due to their RPM

Larger propellers consume more power and create a bigger pressure difference. Put simply, they generate more thrust.

They require low KV motors as they need more torque to move the large blades

Pitch

The pitch is a crucial factor to consider as it's an important part of propeller specifications.



It's defined as the travel distance of the propeller per revolution and is generally measured in inches.

Propellers with a lower pitch spin faster which facilitates higher acceleration. At the same time, less current is drawn.

A higher pitch, on the other hand, means more thrust will be generated. This results in higher with the drawback of more power consumption and additional torque required to drive it. Also high pitch can result on prop stall fluttering which can be dangerous at certain RPM use with caution

Again, the ideal pitch length depends on your drone's application.

50453 or 5x45x3 Prop size in Inch Pitch Angle Number of Blades

Motor & Propeller Selection

For beginners its easy to mismatch these components that can result in unwanted behavior or poor performance of the drone we have a few recommendation what we were tested to work this basic over view chart can show clearly shows the relation ship of Motor to Propeller ratio

Motor KV	Propeller Size and Pitch	Hot	Under Load	Good	Drone Weight & Size Range Type Quad
850KV on 3s	10x45 (1045)				2kg-2.5kg 450mm-550mm
950KV on 3s	10x45 (1045)				2kg-2.5kg 450mm-550mm
920KV on 4s	10x45 (1045) 9x45 (9450) 8x45 (8045 or 8450)				2kg-2.5kg 450mm-550mm
920KV on 3s	10x45 (1045) 9x45 (9450) 8x45 (8045 or 8450)				2kg-2.5kg 450mm-550mm
1300kV on 3s	8x45 (8045 or 8450) 8x3.8 (8038 or 8380)				600g-1kg 330mm-450mm
1700kV on 3s	8x45 (8045 or 8450) 8x3.8 (8038 or 8380)				600g-1kg 330mm-450mm
2000Kv on 3s	6x45x3 (6045 3) 5x45x3 (5045 3)				300g-500g 300mm-330mm
2300kv on 3s	6x45x3 (6045 3) 5x45x3 (5045 3) 5x40x3 (5040 3)				300g-500g 250mm-330mm
2300Kv on 4s	5x45x3(5045 3) 5x45x3 (5040 3)				300g-500g 250mm-330mm
2400KV - 2500kv on 3s	5x45x3 (5045 3) 5x45x3 (5040 3) 4x45x3 (4045 3)				300g-500g 250mm-330mm

Note : Synerduino Drone tend to be slightly heavier than a similar size drone requiring stiffer Props to perform better We recommend 5045 3 or 5040 3



Once all the hardware been calibrated tune and setup you may zip tie the ESC and Tape insulate all the Solder Pads to prevent shorting



ARDUINO IDE

Application Needed

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



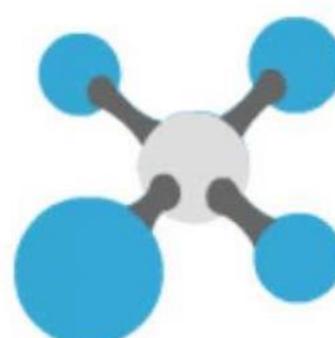
<http://synerflight.com/flywiigui/>

FLYWII GUI & Arduino Drone Multiwii firmware



UBLOX Configuration Platform

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



XCTU Configuration Platform

MULTIWII.INO



Multiwii Sketch

source code for Synerduino KWAD Shield: compile this using the arduino IDE then flash it to your Arduino board

(Arduino 1.8.16) SynerduinoKwad4-GY801-GY91-1.8.16

[Download](#)

()Arduino 1.8.16 – 2.0.0 SynerduinoGY91-1.8.6-2.0.0

[Download](#)

(For UNO boards)Synerduino-Uno-GY91-801

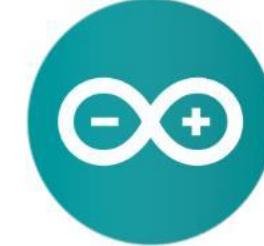
[Download](#)

<http://synerflight.com/kwad-documentation/3-software/>

This is going to be the major bulk of this project down load and open the Multiwii.Ino as in Arduino IDE

This houses the bulk of that the Arduino Drones codes

TYPE OF MULTICOPTER



CONFIG.H

MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

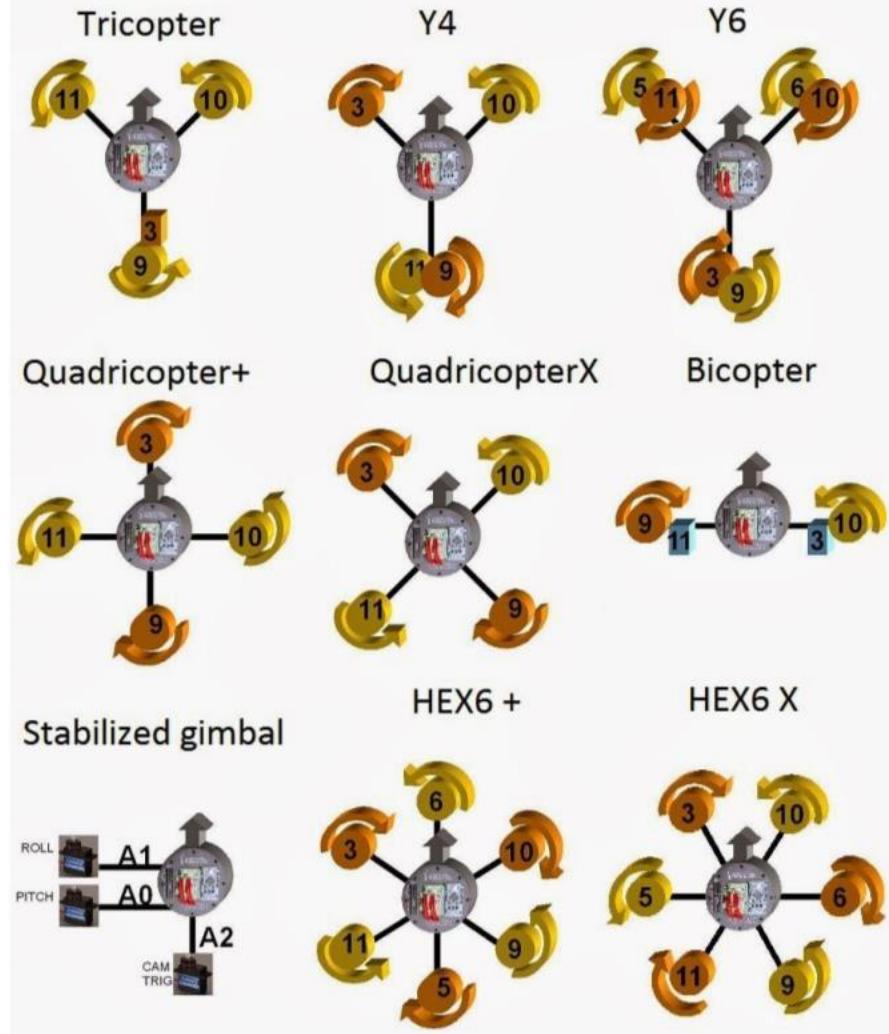
Multivli Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h Multivli.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp

```
/*
 **** The type of multicopter ****
 
 #define GIMBAL
 #define BI
 #define TRI
 #define QUADP
 #define QUADX
 #define Y4
 #define Y6
 #define HEX6
 #define HEX6X
 #define HEX6H // New Model
 #define OCTOX8
 #define OCTOFLATP
 #define OCTOFLATX
 #define FLYING_WING
 #define VTAIL4
 #define AIRPLANE
 #define SINGLECOPTER
 #define DUALCOPTER
 #define HELI_120_CCPM
 #define HELI_90_DEG
 

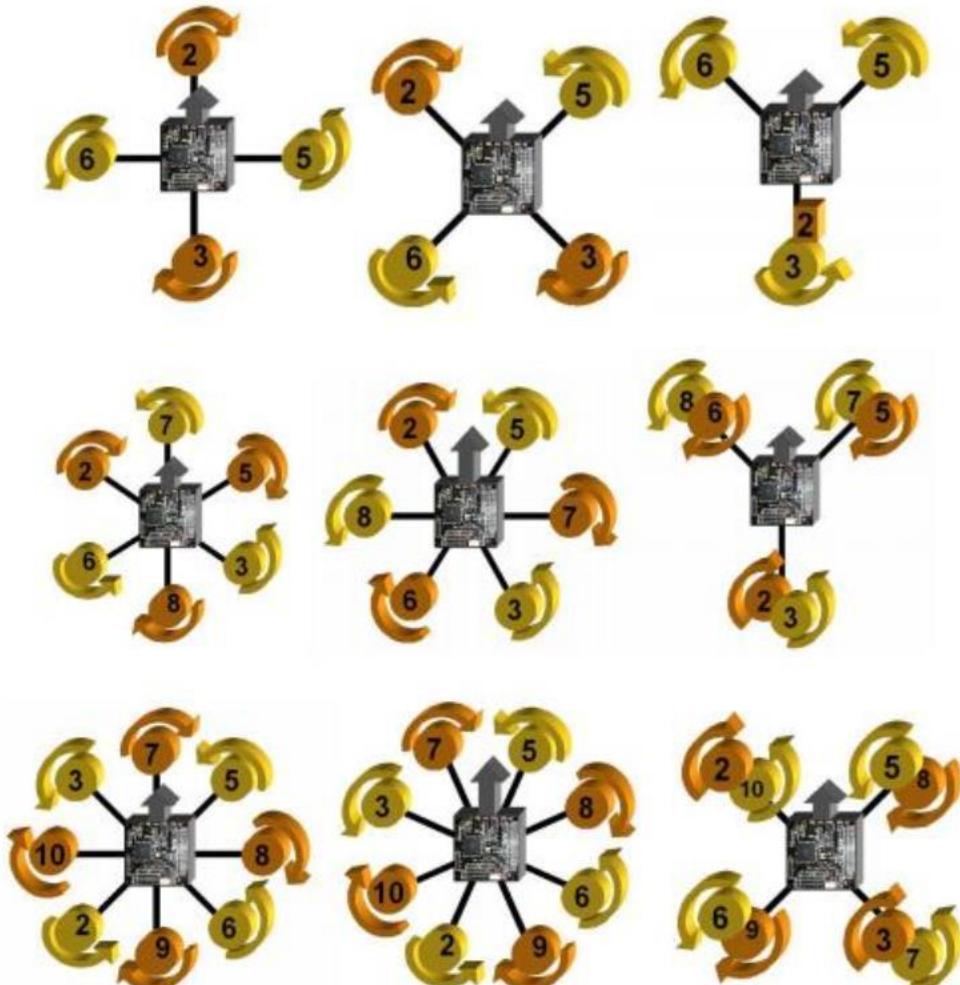
 **** Motor minthrottle ****
 /* Set the minimum throttle command sent to the ESC (Electronic Speed Controller)
 This is the minimum value that allow motors to run at a idle speed */
 #define MINTHROTTLE 1300 // for Turnigy Plush ESCs 10A

Arduino/Genuino Uno on COM41
R A S E N D I O N G 8:54 PM
13/02/2020 2
```

PWM Pins arrangement D2-D10 / PWM Output



UNO 328



MEGA 2560

MIX TABLE



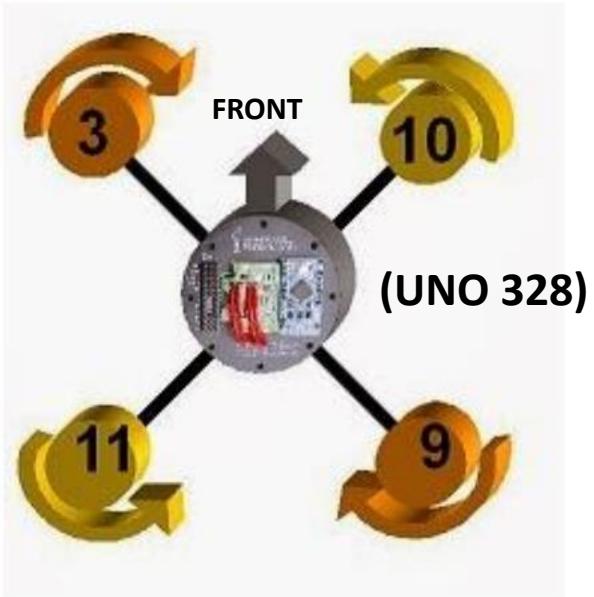
OUTPUT.CPP

```
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.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp lens
/main Mix Table
#ifndef MY_PRIVATE_MIXING
#include "MY_PRIVATE_MIXING"
#endif
#if defined( BI )
motor[0] = PIDMIX(+1, 0, 0); //LEFT
motor[1] = PIDMIX(-1, 0, 0); //RIGHT
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
#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
motor[3] = PIDMIX(+1,-1, 0); //FRONT_L_CW
#endif
1157 - 1153
Arduino/Genuino Uno on COM4
9:01 PM 13/02/2020
```

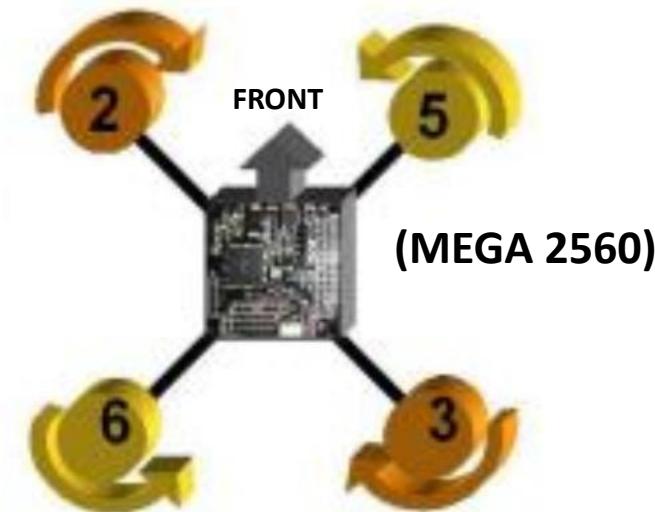
Mix Table shows the motors setup on input this is also helpful for those custom airframe designs require special mixing

OUTPUT.CPP

Motor [3]



Motor [2]



Motor [3]

Motor [1]

Motor [0]

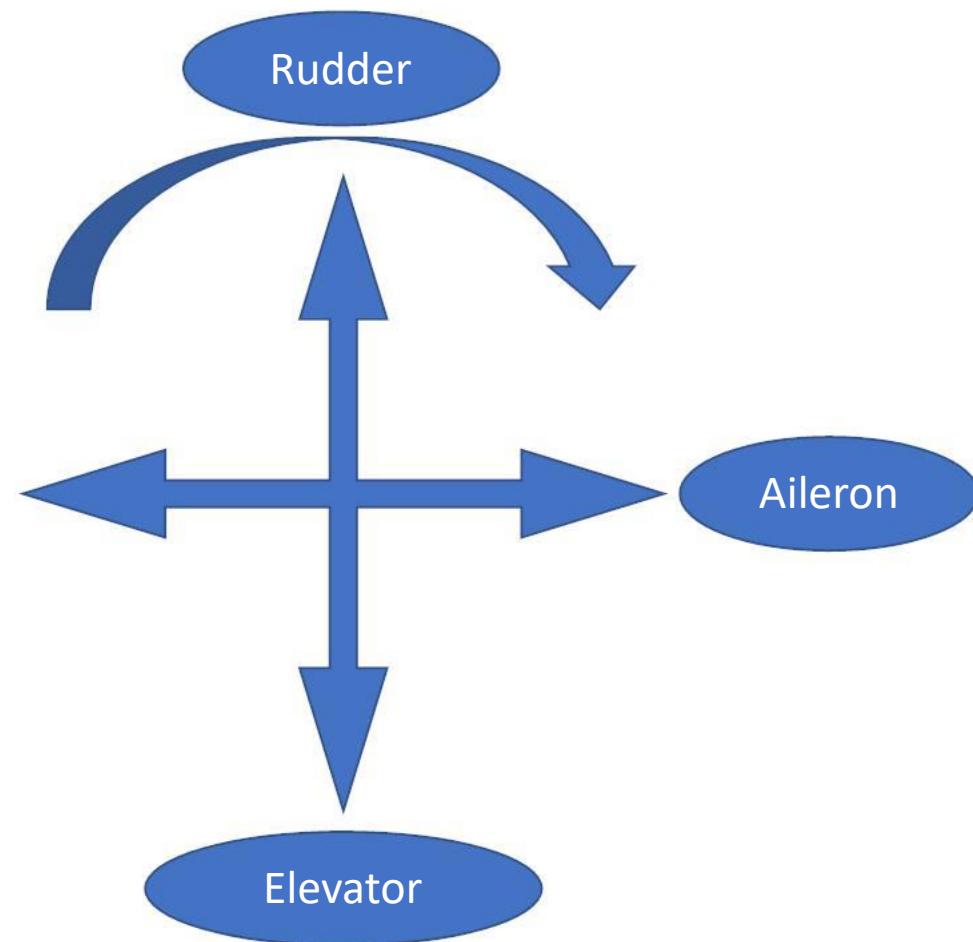
Motor [1]

Motor [0]

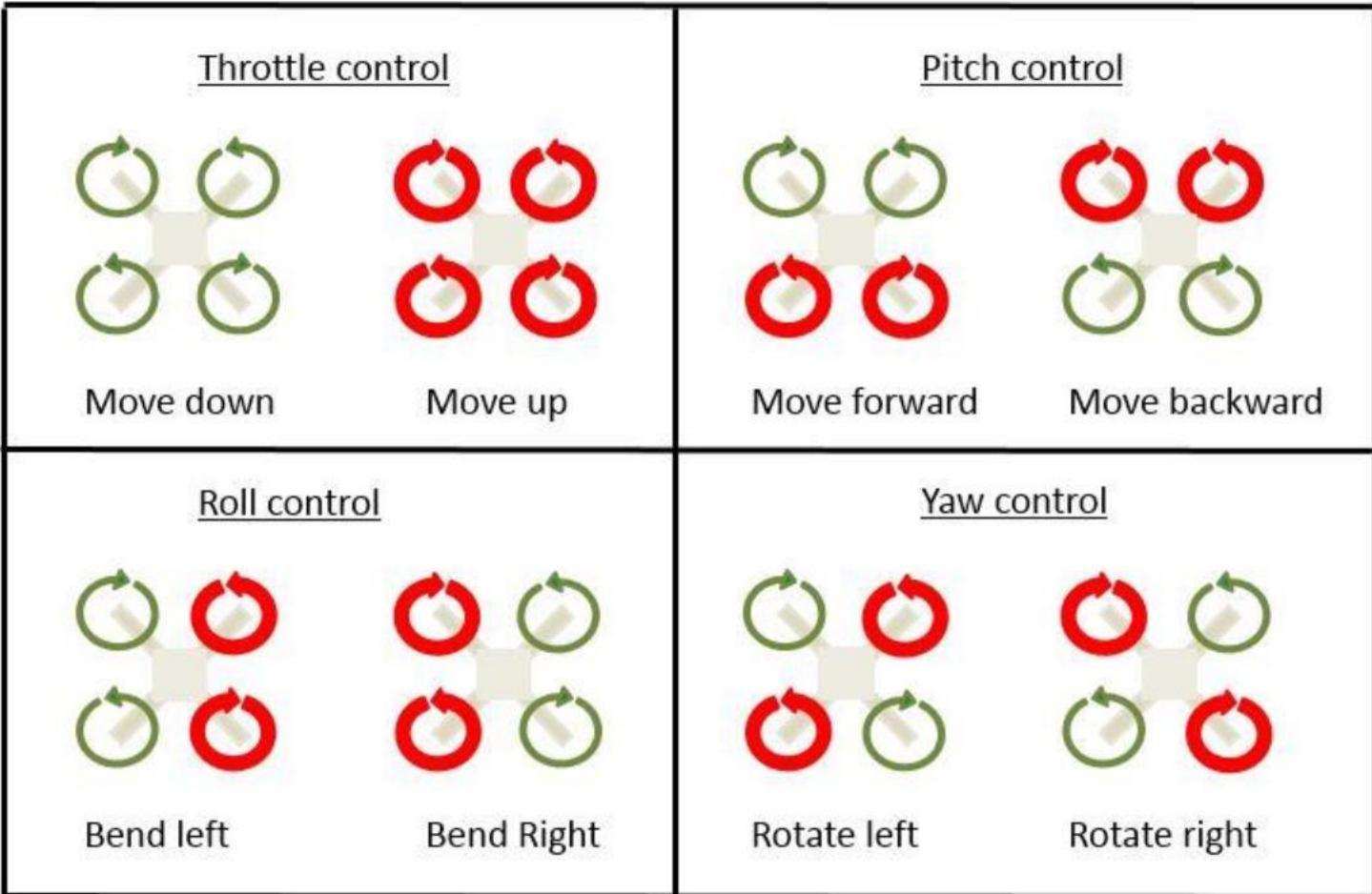
Roll Right

Pitch Forward

Yaw Right



MOTOR MIX FOR QUAD



 Normal Speed (-)

 High Speed (+)

Motor Min Throttle

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

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

```
/*#define 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 */
#ifndef MINTHROTTLE
#define MINTHROTTLE 1300 // for Turnigy Plush ESCs 10A
#define MINTHROTTLE 1120 // for Super Simple ESCs 10A
#define MINTHROTTLE 1064 // special ESC (simonk)
#define MINTHROTTLE 1050 // for brushed ESCs like ladybird
#define MINTHROTTLE 1150 // (*) (**)

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

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

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

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

***** constant loop time *****/

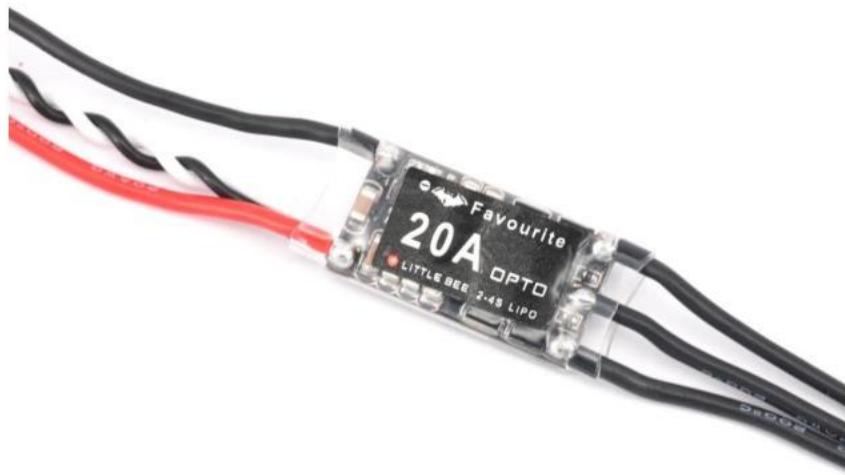
```

63 Arduino/Genuino Uno on COM41

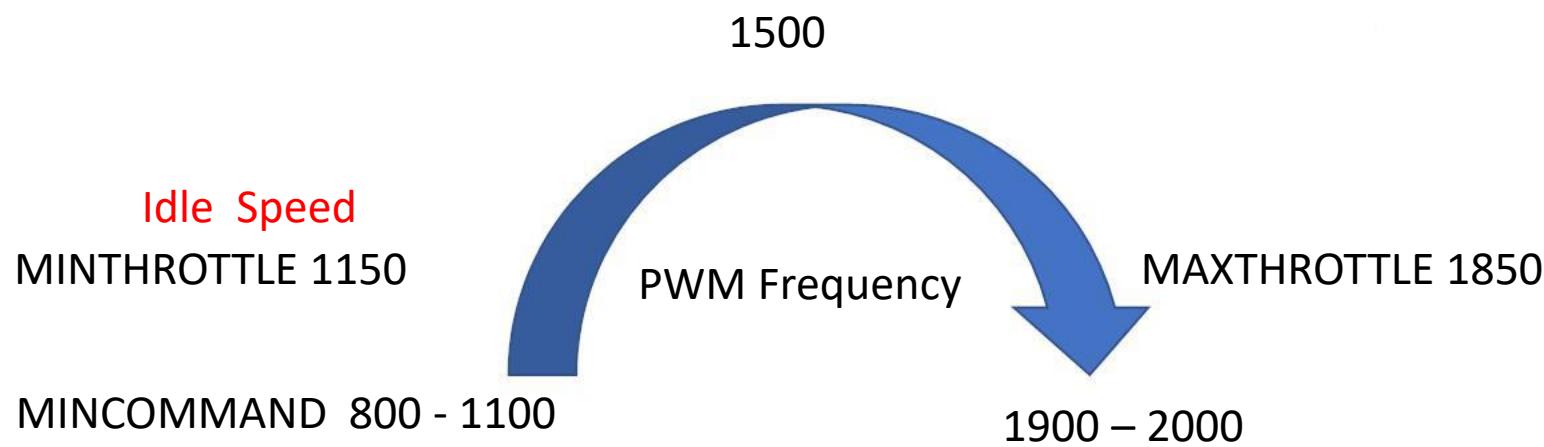
9:20 PM ENG 13/02/2020

Motor Min Throttle

Electronic Speed Controller



Brushless Motor



RECEIVER TYPES



PPM RECEIVER

SBUS RECEIVER



PWM RECEIVER



RECEIVER TYPES

CONFIG.H



For PPM Receiver

Channel Mapping

Uncomment PPM on
Throttle

Pin A8 for Mega
Pin D2 for Uno

You may need to
uncomment and
change the ordering
of your channel
depending on your
Transmitter's model
and specification

```
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 lens

At this moment you can use this function only with WinGUI 2.3 release. MultiWiiConf does not support it yet
*/
#ifndef EXTENDED_AUX_STATES

/*
***** special receiver types *****
***** PPM Sum Receiver *****

/* The following lines apply only for specific receiver with only one PPM sum signal, on digital PIN 2
Select the right line depending on your radio brand. Feel free to modify the order in your PPM order is different */
#define SERIAL_SUM_PPM PITCH,YAW,THROTTLE,ROLL,AUX1,AUX2,AUX3,AUX4,8,9,10,11 //For Graupner/Spektrum
#define SERIAL_SUM_PPM ROLL,PITCH,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11 //For Robe/Hitec/Futaba
#define SERIAL_SUM_PPM ROLL,PITCH,YAW,THROTTLE,AUX1,AUX2,AUX3,AUX4,8,9,10,11 //For Multiplex
#define SERIAL_SUM_PPM PITCH,ROLL,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11 //For some Hitec/Sanwa/Others

// Uncommenting following line allow to connect PPM_SUM receiver to standard THROTTLE PIN on MEGA boards (eg. A8 in CRIUS AIO)
#define PPM_ON_THROTTLE

***** Spektrum Satellite Reciever *****
/* The following lines apply only for Spektrum Satellite Receiver
Spektrum Satellites are 3V devices. DO NOT connect to 5V!
For MEGA boards, attach sat grey wire to RX1, pin 19. Sat black wire to ground. Sat orange wire to Mega board's 3.3V (or any other 3V to 3.3V source).
For PROMINI, attach sat grey to RX0. Attach sat black to ground. */
#define SPEKTRUM_1024
#define SPEKTRUM_2048
#define RX_SERIAL_PORT 1 // Forced to 0 on Pro Mini and single serial boards; Set to your choice of 0, 1, or 2 on any Mega based board (defaults to 1 on Mega).

366
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM30
^ ENG 11:14 AM 11/01/2021
```

RECEIVER TYPES

CONFIG.H



For SBUS Receiver

Channel Mapping

Uncomment SBUS
on RX Serial Port 1
(Telemetry 1)

You may need to
uncomment and
change the ordering
of your channel
depending on your
Transmitter's model
and specification

```
 392: //*****
 393: // Defines that allow a "Bind" of a Spektrum or Compatible Remote Receiver (aka Satellite) via Configuration GUI.
 394: // Bind mode will be same as declared above, if your TX is capable.
 395: // Ground, Power, and Signal must come from three adjacent pins.
 396: // By default, these are Ground=4, Power=5, Signal=6. These pins are in a row on most MultiWii shield boards. Pins can be overriden below.
 397: // Normally use 3.3V regulator is needed on the power pin!! If your satellite hangs during bind (blinks, but won't complete bind with a solid light), go direct 5V on all pins.
 398: //*****
 399: // For Pro Mini, the connector for the Satellite that resides on the FTDI can be unplugged and moved to these three adjacent pins.
400: //#define SPEK_BIND           //Un-Comment for Spektrum Satellie Bind Support. Code is ~420 bytes smaller without it.
401: //#define SPEK_BIND_GROUND 4
402: //#define SPEK_BIND_POWER   5
403: //#define SPEK_BIND_DATA    6
404:
405: //***** SBUS RECEIVER *****/
406: /* The following line apply only for Futaba S-Bus Receiver on MEGA boards or PROMICRO boards.
407: You have to invert the S-Bus-Serial Signal e.g. with a Hex-Inverter like IC SN74 LS 04 */
408: //#define SBUS   PITCH,YAW,THROTTLE,ROLL,AUX1,AUX2,AUX3,AUX4,8,9,10,11,12,13,14,15,16,17 // dsm2 orangerx
409: #define SBUS   ROLL,PITCH,THROTTLE,YAW,AUX1,AUX2,AUX3,AUX4,8,9,10,11,12,13,14,15,16,17 // T14SG
410: #define RX_SERIAL_PORT 1
411: #define SBUS_MID_OFFSET 988 //SBUS Mid-Point at 1500
412:
413: //***** HOTT RECEIVER *****/
414: /* Graupner Hott HD */
415: //#define SUMD PITCH,YAW,THROTTLE,ROLL,AUX1,AUX2,AUX3,AUX4
416: //#define RX_SERIAL_PORT 1
417:
418: //***** SECTION 4 - ALTERNATE CPUs & BOARDS *****/
419: //*****
420: //*****
421: //*****
422: //*****
423:
```

Done Saving.
avrdude done. Thank you.

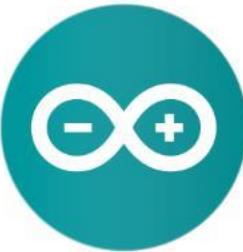
410-409

Arduino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM17

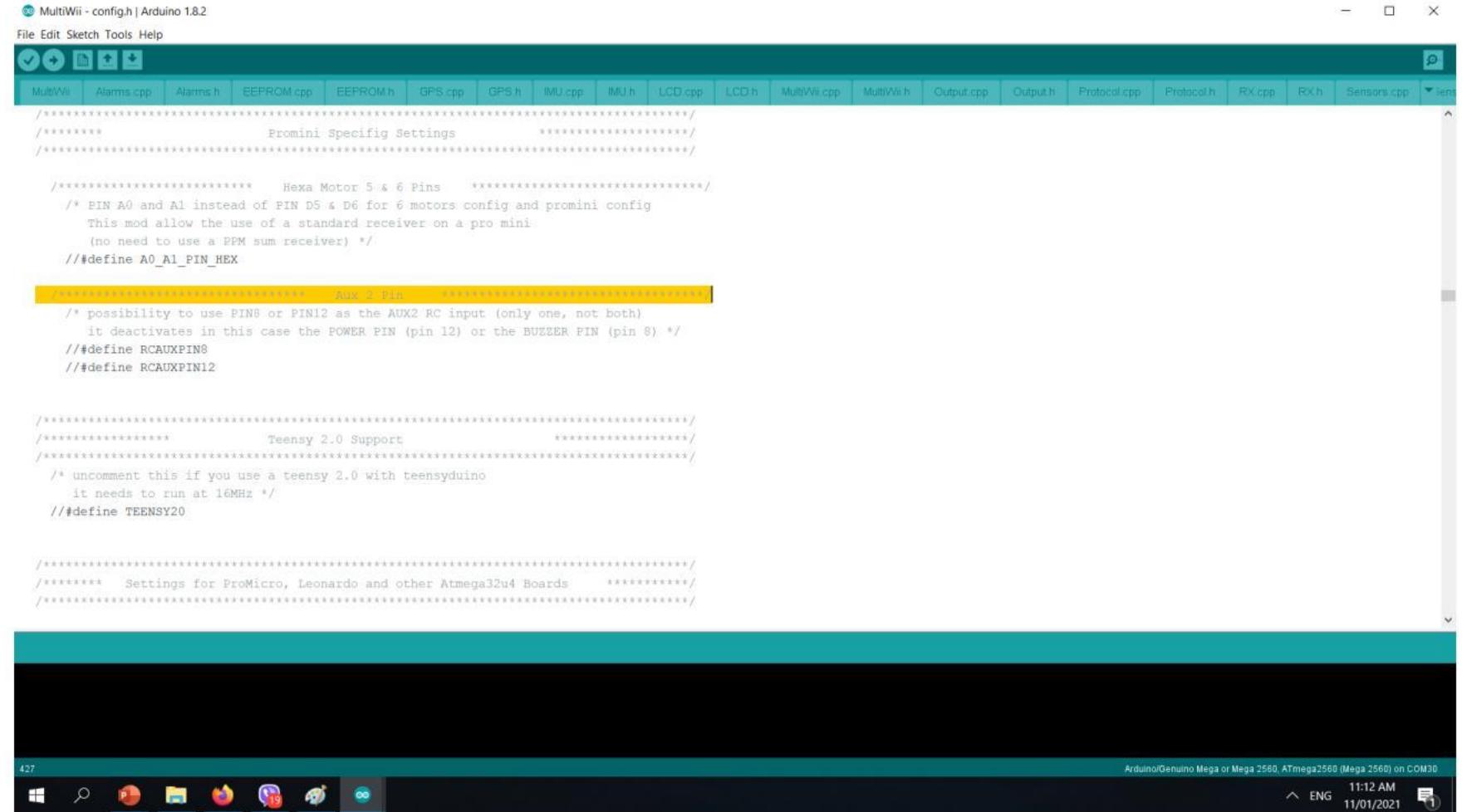
Links ENG 8:48 PM 24/07/2022

AUX PIN

CONFIG.H



For UNO you need to
define your Aux 2
Pin



```
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

//*****
//*****          Promini Specific Settings          *****
//*****

//*****
//*****          Hexa Motor 5 & 6 Pins          *****
//** PIN A0 and A1 instead of PIN D5 & D6 for 6 motors config and promini config
// This mod allow the use of a standard receiver on a pro mini
// (no need to use a PPM sum receiver)
// #define A0_A1_PIN_HEX

//*****
//*****          AUX 2 PIN          *****
// * possibility to use PIN8 or PIN12 as the AUX2 RC input (only one, not both)
// it deactivates in this case the POWER PIN (pin 12) or the BUZZER PIN (pin 8)
// #define RCAUXPIN8
// #define RCAUXPIN12

//*****
//*****          Teensy 2.0 Support          *****
//*****
// * uncomment this if you use a teensy 2.0 with teensyduino
// it needs to run at 10MHz
// #define TEENSY20

//*****
//*****          Settings for ProMicro, Leonardo and other Atmega32u4 Boards          *****
//*****
```

Arduino/Genuino Mega or Mega 2560, ATMega2560 (Mega 2560) on COM30
11:12 AM
11/01/2021

INERTIAL MEASURING UNIT MEASURING UNIT

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

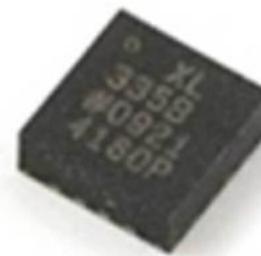


Magnetometer

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 p. Forum. Do not hesitate to add your board here if you have any information about it.
   // confirmed by Alex
   // confirmed by Alex
   // v0.1 & v0.2 & v0.3 version of 9DOF board from Fabio
   // FreeIMU v0.3 and v0.3.1
   // FreeIMU v0.3.5 no baro
   // FreeIMU v0.3.5_MS
   // FreeIMU v0.3.5_BMP
   // FreeIMU v0.4 with MPU6050, HMC5883L, MS561101BA
   // same as FREEIMUV04 with final MPU6050 (with the right ACC scale)
   // the smallest multiwii FC based on MPU6050 + pro micro based proc
   // 9DOF board from erazz
   // full FC board 9DOF+baro board from witespy with BMP085 baro
   // full FC board 9DOF+baro board from witespy second edition
   // full FC board 9DOF+baro board from witespy second edition
   // full FC board or standalone 9DOF+baro board from CSG_EU
   // AEROQUADSHIELDv2
   // Atmel 9DOF (Contribution by EOSBandi). requires 3.3V power.
   // Sirius Navigator IMU
   // Sirius Navigator IMU using external MAG on GPS board
   // confirmed by Alex
   // 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 comment 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

Independent Sensors

```
/* leave it commented if you already checked a specific board above */
/* I2C gyroscope */
#ifndef define WMP
#define ITG3050
#ifndef 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
```

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

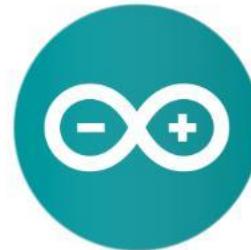
```
#define L3G4200D // gyro
#define ADXL345 // acc
#define BMP085 // baro
#define MMC5883 // mag
```

Synerduino GY91

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

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

SENSORS.CPP



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 sens

```
#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

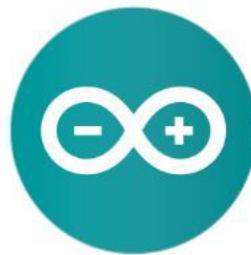
// *****
// I2C Gyroscope ITG3200 / ITG3205 / ITG3050 / MPU3050
<
```

851 Arduino/Genuino Uno on COM41

9:37 PM ENG 13/02/2020

The screenshot shows the Arduino IDE interface with the file "Sensors.cpp" open. The code implements I2C communication with a gyroscope sensor, specifically the L3G4200D. It includes functions for initializing the sensor and reading six raw ADC values. The Arduino Uno is connected to port COM41, and the system tray shows the date and time as 9:37 PM on 13/02/2020.

SENSORS.CPP



MultiWii - Sensors.cpp | Arduino 1.8.2

File Edit Sketch Tools Help

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

```
// 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);

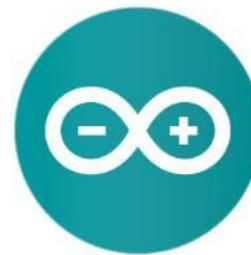
    ACC_ORIENTATION( ((rawADC[1]<<8) | rawADC[0]) ,
                    ((rawADC[3]<<8) | rawADC[2]) ,

```

642 Arduino/Genuino Uno on COM41

9:38 PM 13/02/2020

SENSORS.CPP



MultiWii - Sensors.cpp | 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.h MultiWii.cpp Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp lens

```
// ****
// 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

R A ^ S D ENG 9:40 PM 13/02/2020 2

SENSORS.CPP



MultiWii - Sensors.cpp | Arduino 1.8.2
File Edit Sketch Tools Help

Sensors.cpp

```
// ****
// I2C_Compas_HMC5883
// ****
// I2C adress: 0x3C (8bit) 0x1E (7bit)
// ****

#if defined(HMC5883)

#define HMC58X3_R_CONF_A 0
#define HMC58X3_R_CONF_B 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.
```

997 Arduino/Genuino Uno on COM41 9:43 PM 13/02/2020

SENSOR ORIENTATION

CONFIG.H



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

MultWii 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

```
//#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 phisically for flying in X mode (or vice versa)
 * you can use one of this options for virtual sensors rotation by 45 degress, 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 degress clockwise
#define SENSORS_TILT_45DEG_LEFT // rotate the FRONT 45 degress 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

SERIAL COM

CONFIG.H



SERIAL BAUD RATE

WILL DEPEND ON WHAT BAUD YOUR
TELEMETRY MODULE ARE SET INTO
YOU CAN CHANGE IT TO SUITE THE
PORT YOUR DEVICE IS CONNECTED
TO.

SERIAL 0 CAN BE USE FOR TELEMETRY
GIVEN NOTHING IS CONNECTED TO
THE USB AT THIS POINT. AND
FIRMWARE MUST BE FLUSH PRIOR TO
HOOKING UP ANYTHING TO THIS PINS

SERIAL 1 , 3 IS RESERVE FOR
TELEMETRY

115200 FOR BLUETOOTH HC-05

38400 FOR XBEE RADIO

57600 for SIK RADIO

SERIAL 2 IS RESERVE FOR GPS

NMEA Baud 57600

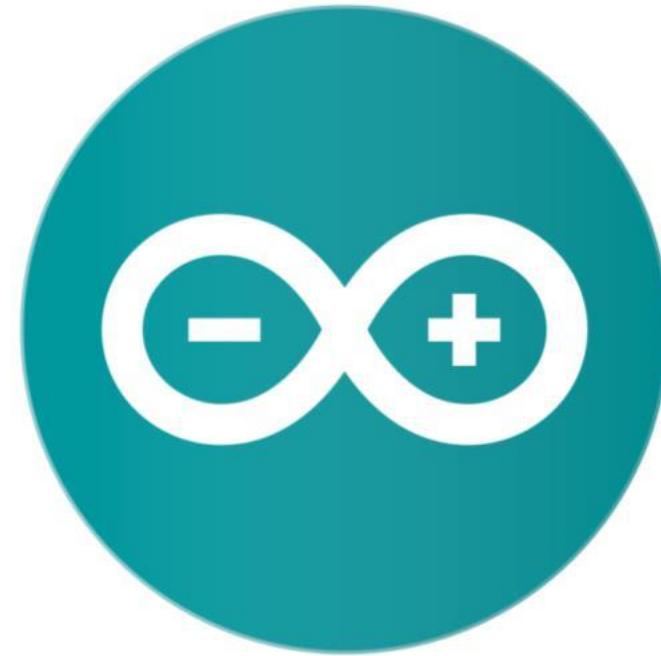
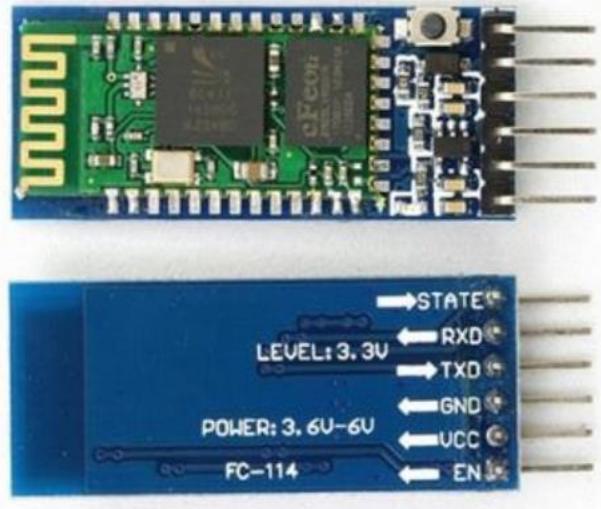
```
MultiWii - config.h | Arduino 1.8.2
File Edit Sketch Tools Help
MultiWii Alarms.cpp Alarms.h EEPROM.cpp EEPROM.h GPS.cpp GPS.h IMU.cpp IMU.h LCD.cpp LCD.h MultiWii.cpp MultiWii.h Output.cpp Output.h Protocol.cpp Protocol.h RX.cpp RX.h Sensors.cpp send

/*
***** 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.
```

Arduino/Genuino Uno on COM4 9:47 PM 13/02/2020



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=ArduinoDrone

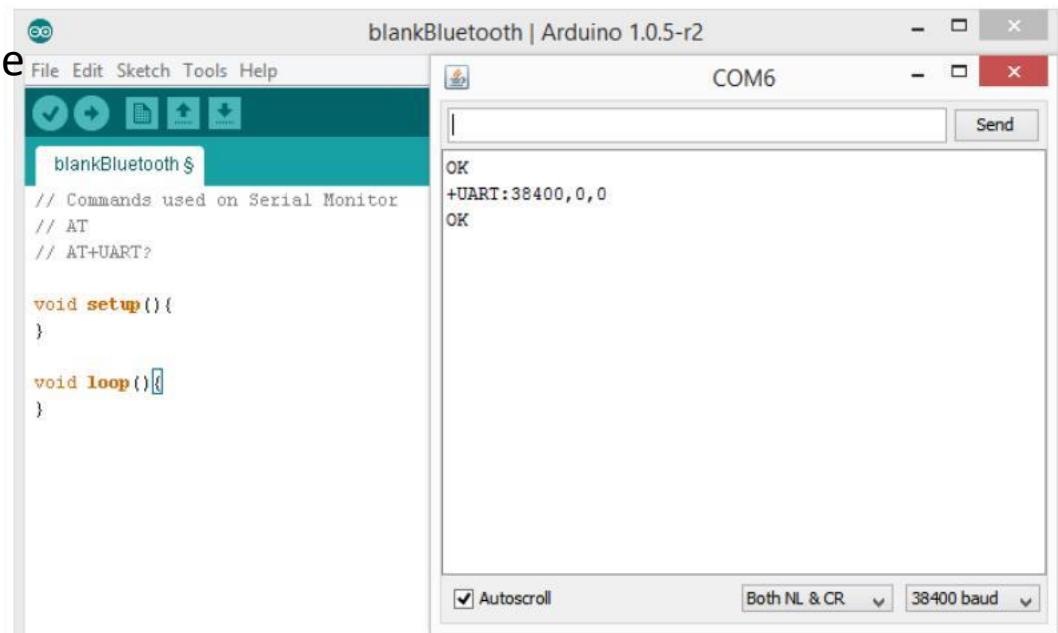
AT+PSWD=1234 (Version 2)

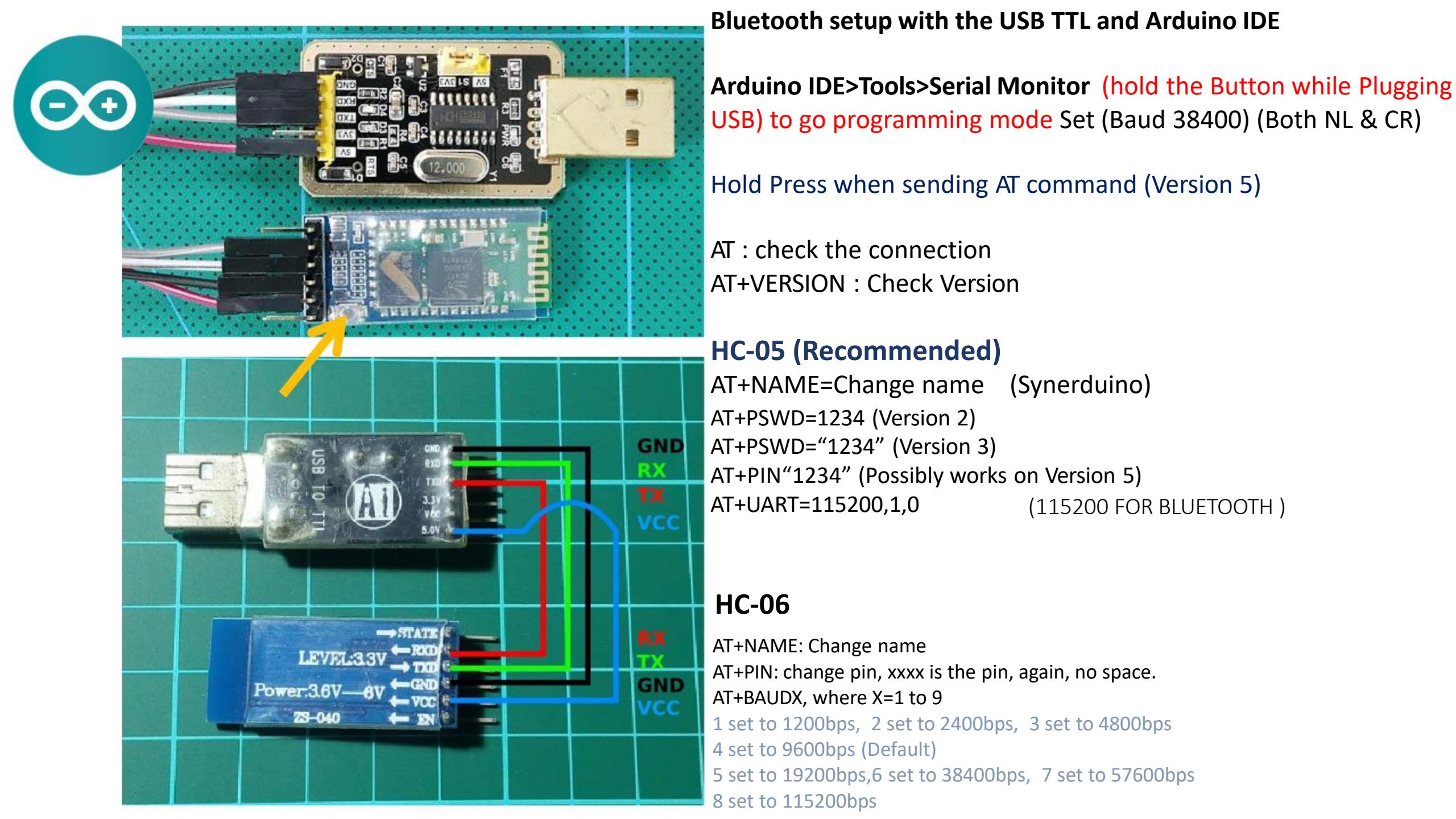
AT+PSWD="1234" (Version 3)

AT+UART=115200,1,0



HC-05 (Recommended)





Bluetooth setup with the USB TTL and Arduino IDE

Arduino IDE>Tools>Serial Monitor (hold the Button while Plugging 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)

AT+PIN"1234" (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.

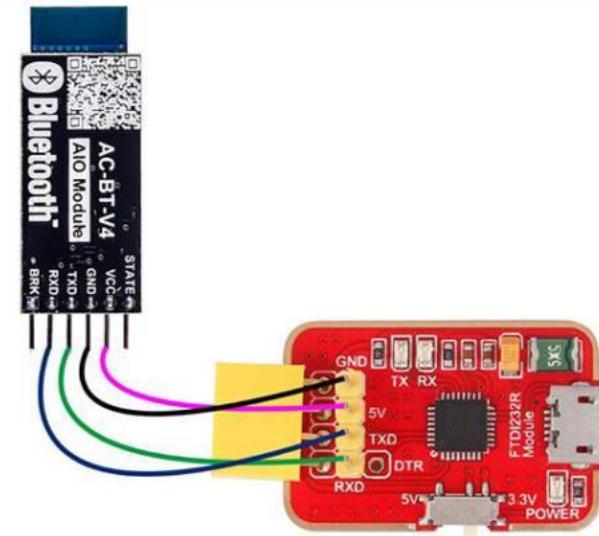
Example

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)

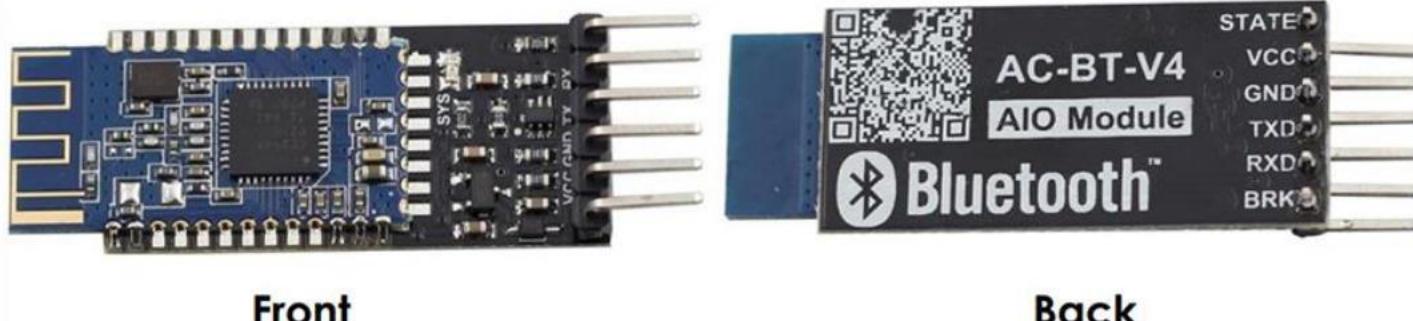
AT+PASS123456 Set password to 123456

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

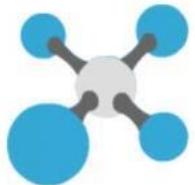


Download

[40003026 CXCTU-1.zip](#)



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

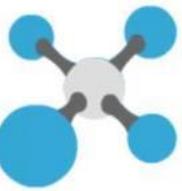
Force the module to maintain its current configuration.

Select current

[View Release Notes](#)

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

GROUND STATION



XCTU

Radio Modules

Name: ZigBee Router AT
Function: 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

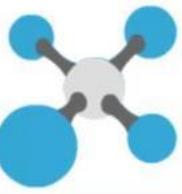
NT Node Discovery Backoff: 3C x 100 ms

AC Node Discovery Outtime:

Checking for Radio Firmware updates: (87%)

2:30 PM 07/04/2020

GROUND STATION



XCTU

Radio Modules

Name: ZigBee Router AT
Function: ZigBee Router AT
Port: COM35 - 38400/8/N/1/N - AT
MAC: 0013A20040A11A91

Radio Configuration [- 0013A20040A11A91]

EE Encryption Enable: Disabled [0] Bitfield

EO Encryption Options: 0 Bitfield

KY Encryption Key: []

Serial Interfacing

Baud Rate: 38400 [5] x character times

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

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

SM Sleep Mode: No Sleep (Router) [0]

SN Number of Cyclic Sleep Periods: 1

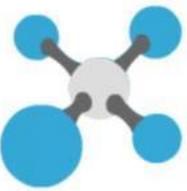
SO Sleep Options: 0

SP Cyclic Sleep Period: 20 x 10 ms

ST Time before Sleep: 1388 x 1 ms

Blue arrow pointing to the Baud Rate field.

AIRCRAFT



XCTU

Radio Modules

Name: ZigBee Coordinator AT
Function: ZigBee Coordinator AT
Port: COM36 - 38400/8/N/1/N - AT
MAC: 0013A2004081A7F

Radio Configuration [- 0013A2004081A7F]

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		

Windows taskbar icons: File Explorer, Firefox, Edge, Task View, File History, Task Scheduler, Control Panel, File Explorer, Task View, File History, Task Scheduler, Control Panel.

System tray: Volume, Network, Battery, ENG, 2:48 PM, 07/04/2020.

SIK SERIAL RADIO

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

[3drradioconfig](#)

[Download](#)

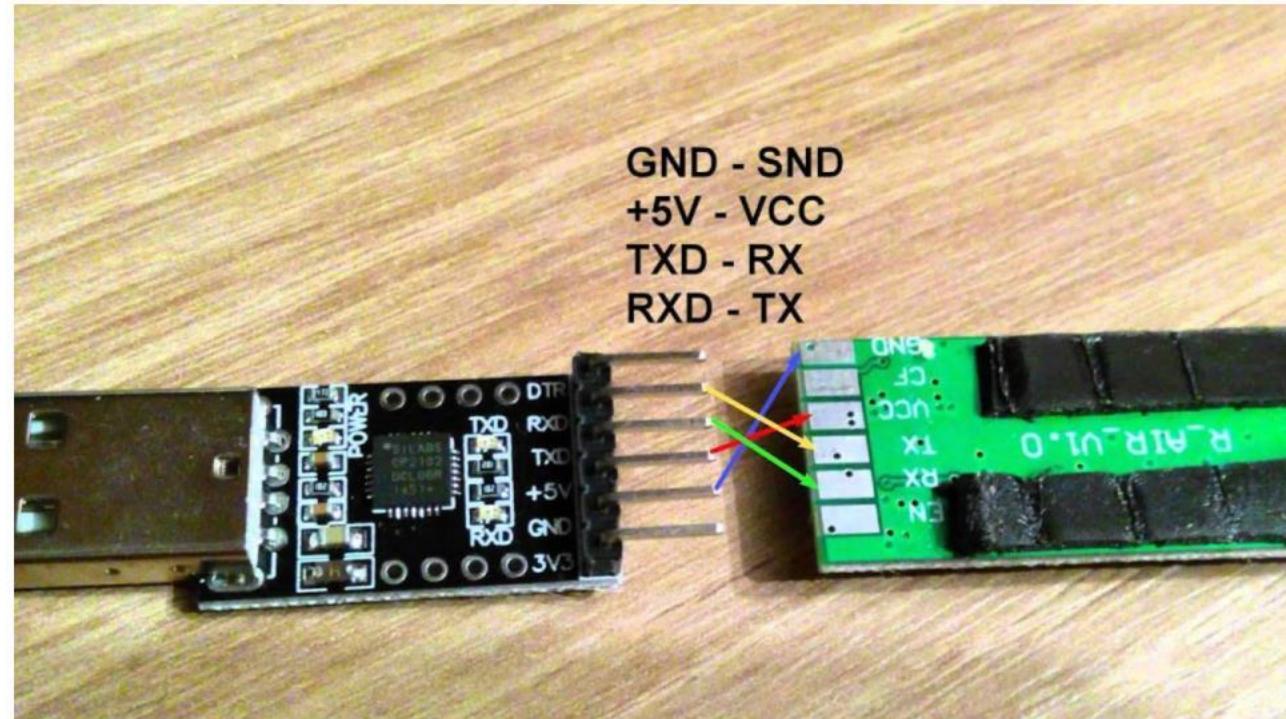


RadioTelemetry Air Module



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 mostlikely 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

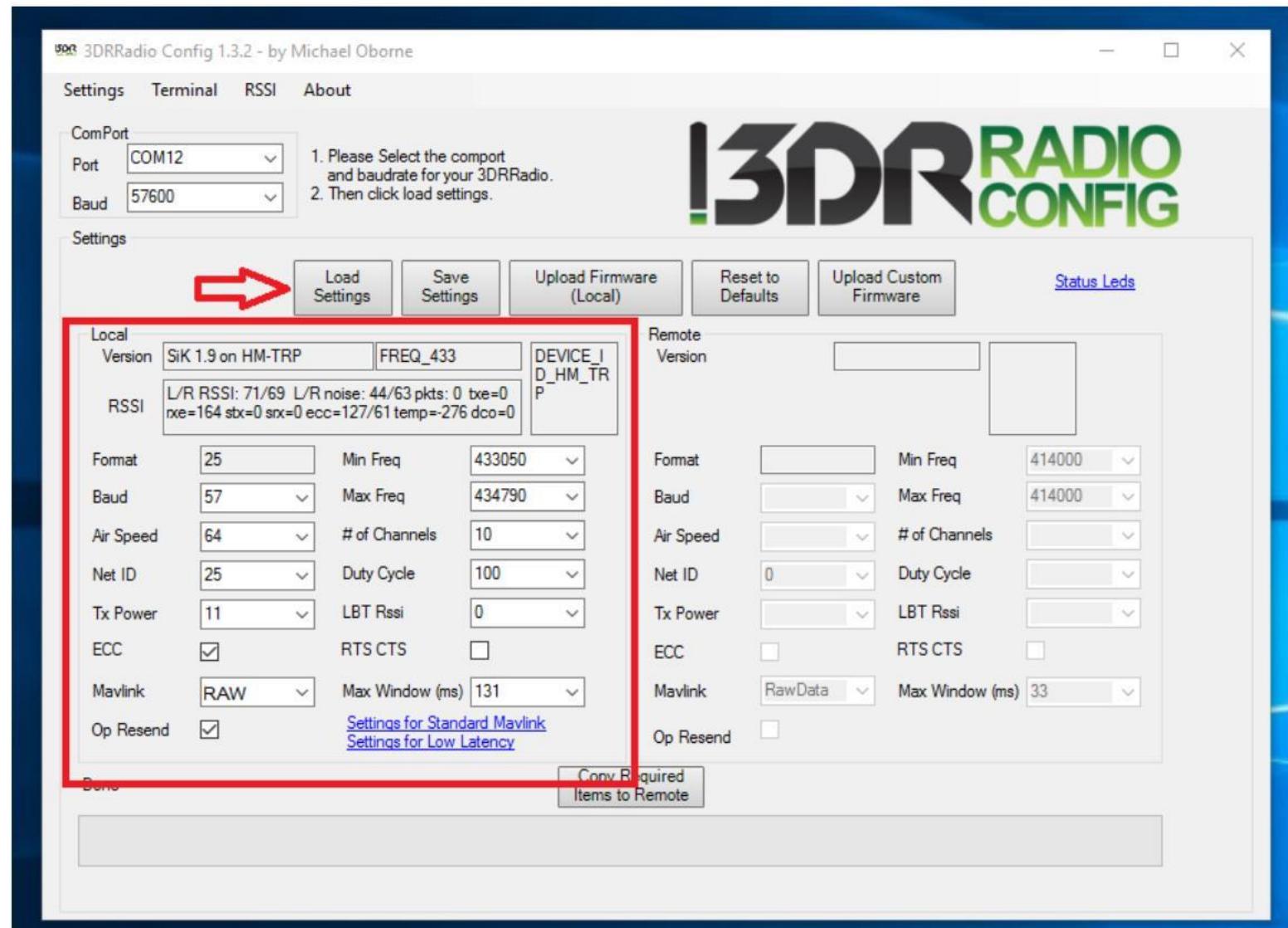
Manually configuring the telemetry kit for Synerduino uses the 3DR radio Config

<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

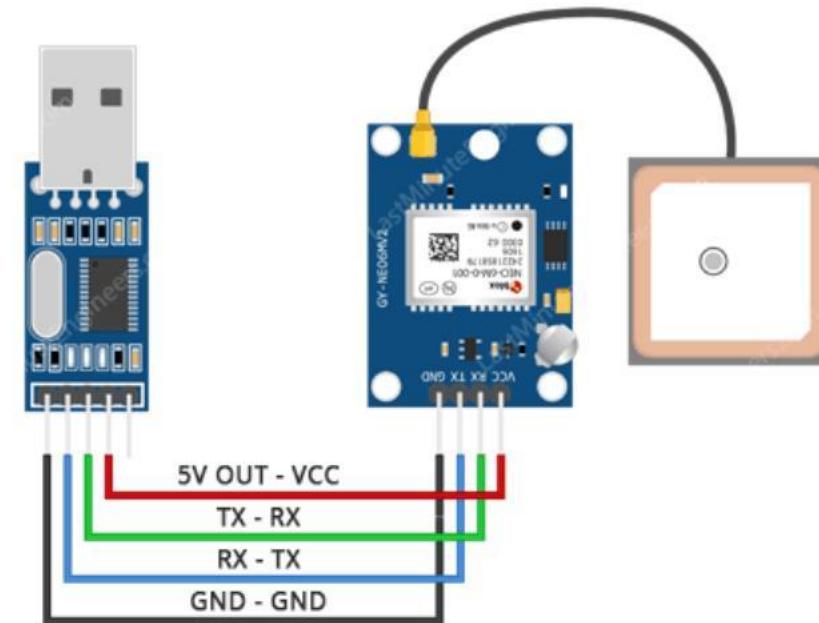
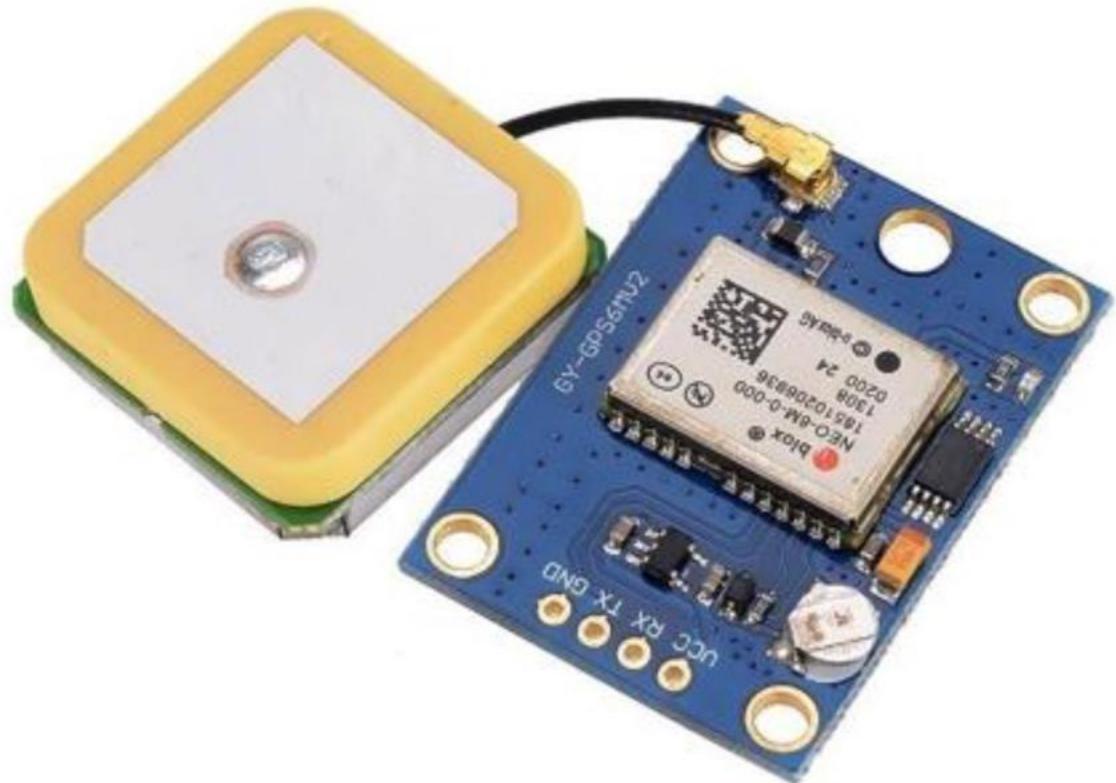
ublox



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

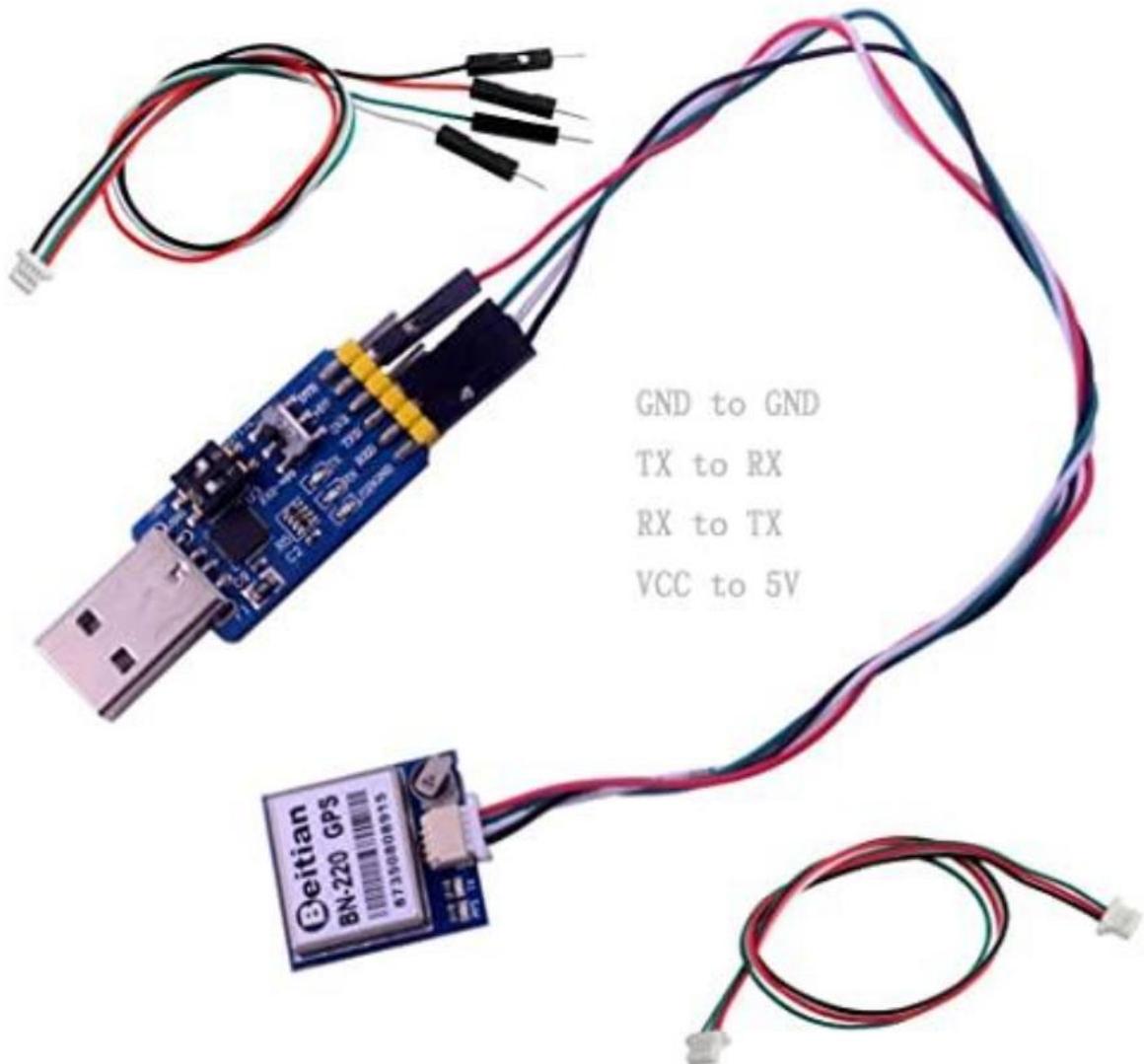


Pin Description:

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

GPS CONFIGURING



BEITIAN UBLOX

PLUG IN TO SERIAL TX 2 RX 2

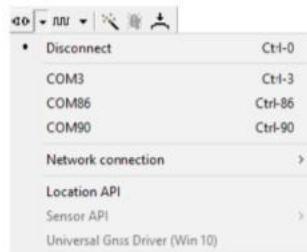
USB TTL TO PROGRAM THE GPS

THIS GOES SAME ON THE DRONE SHIELD

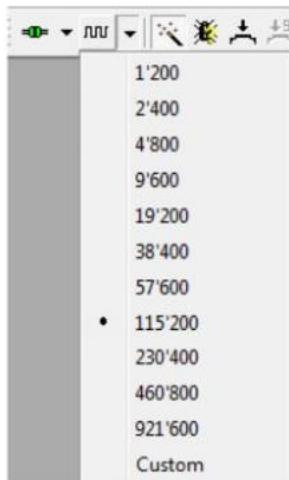
U CENTER



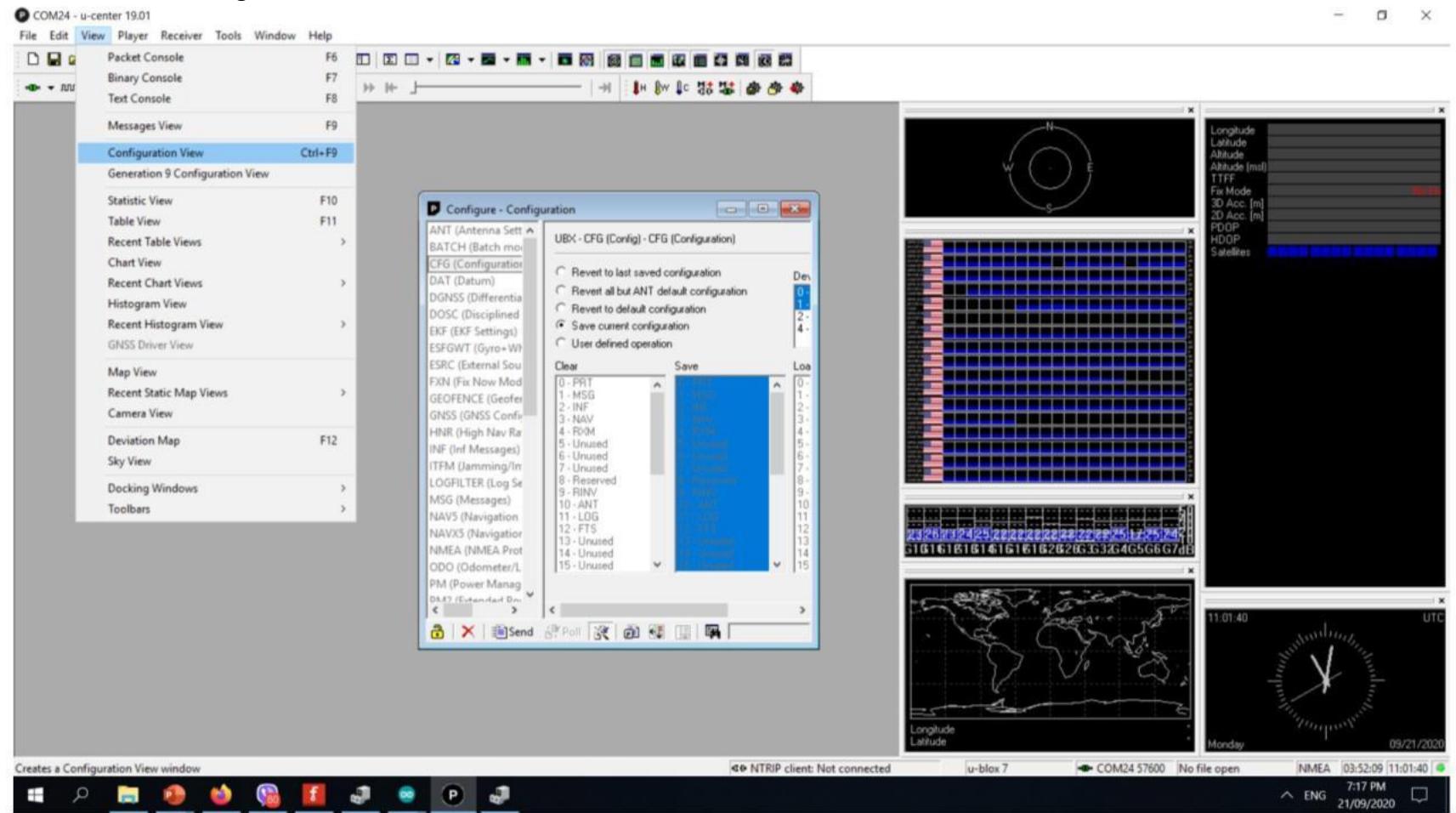
Connect to the device. Select the COM port your GPS and USBTTL is connected to



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



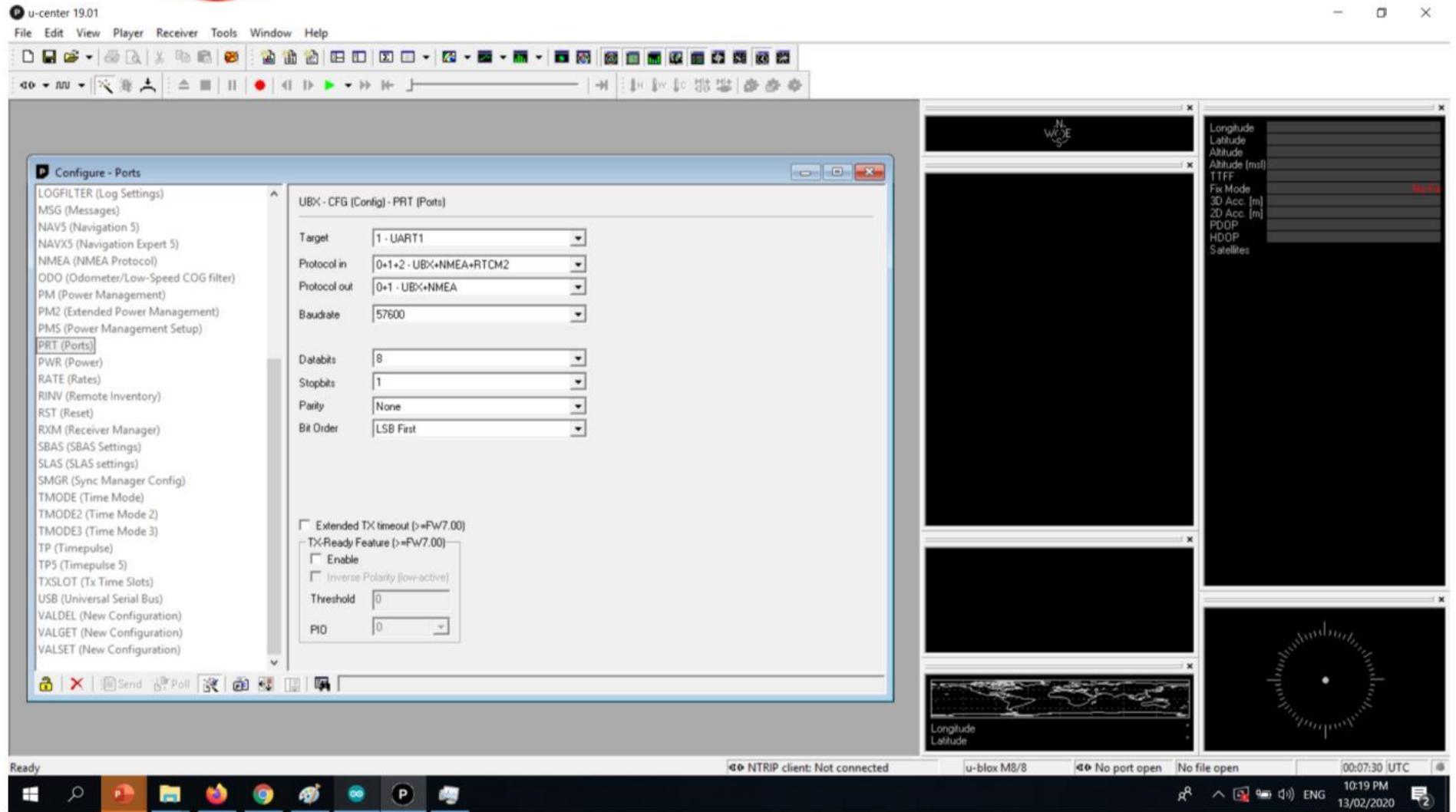
Go to View> Configuration View



U CENTER



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



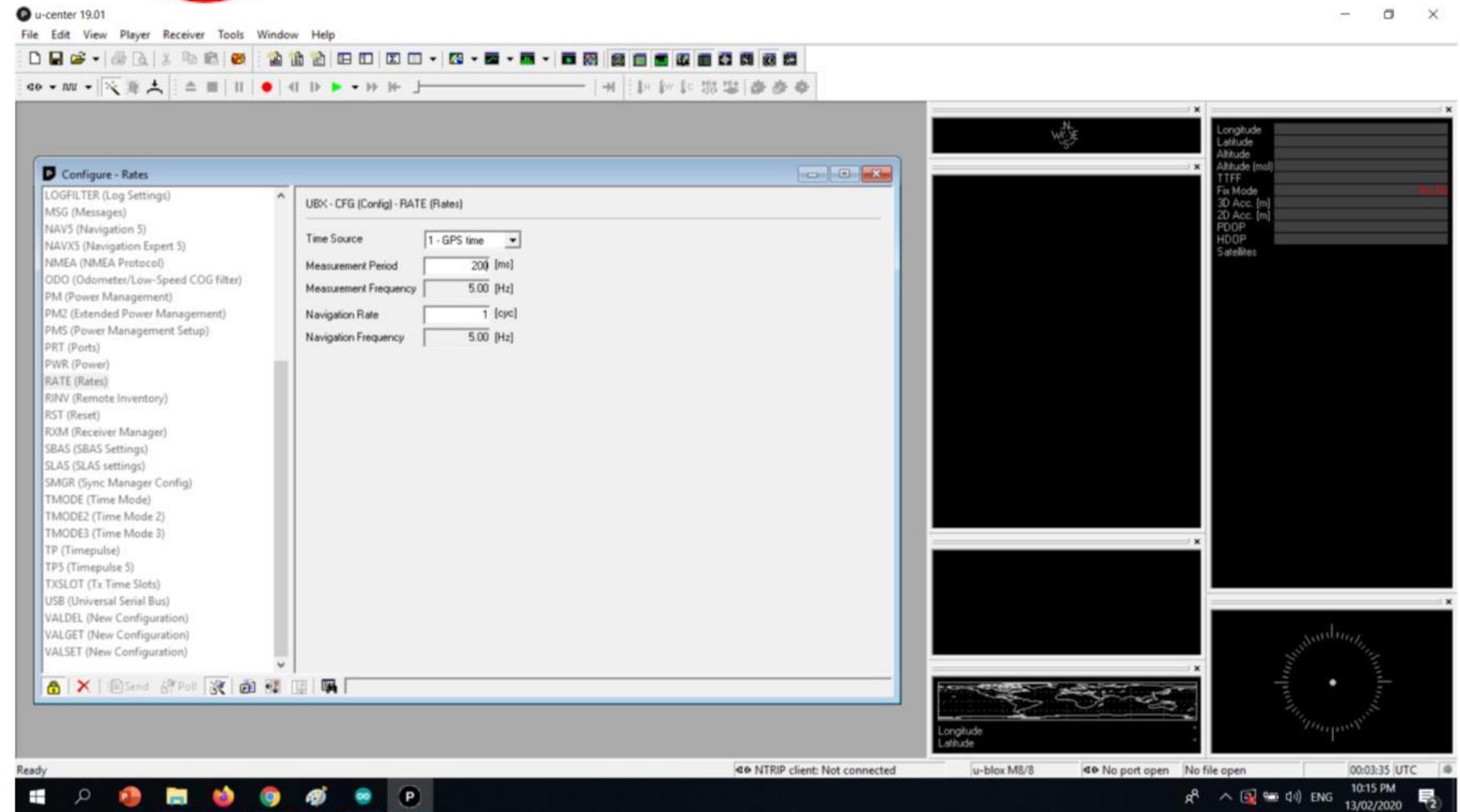
U CENTER



Rate 200ms
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”



U CENTER



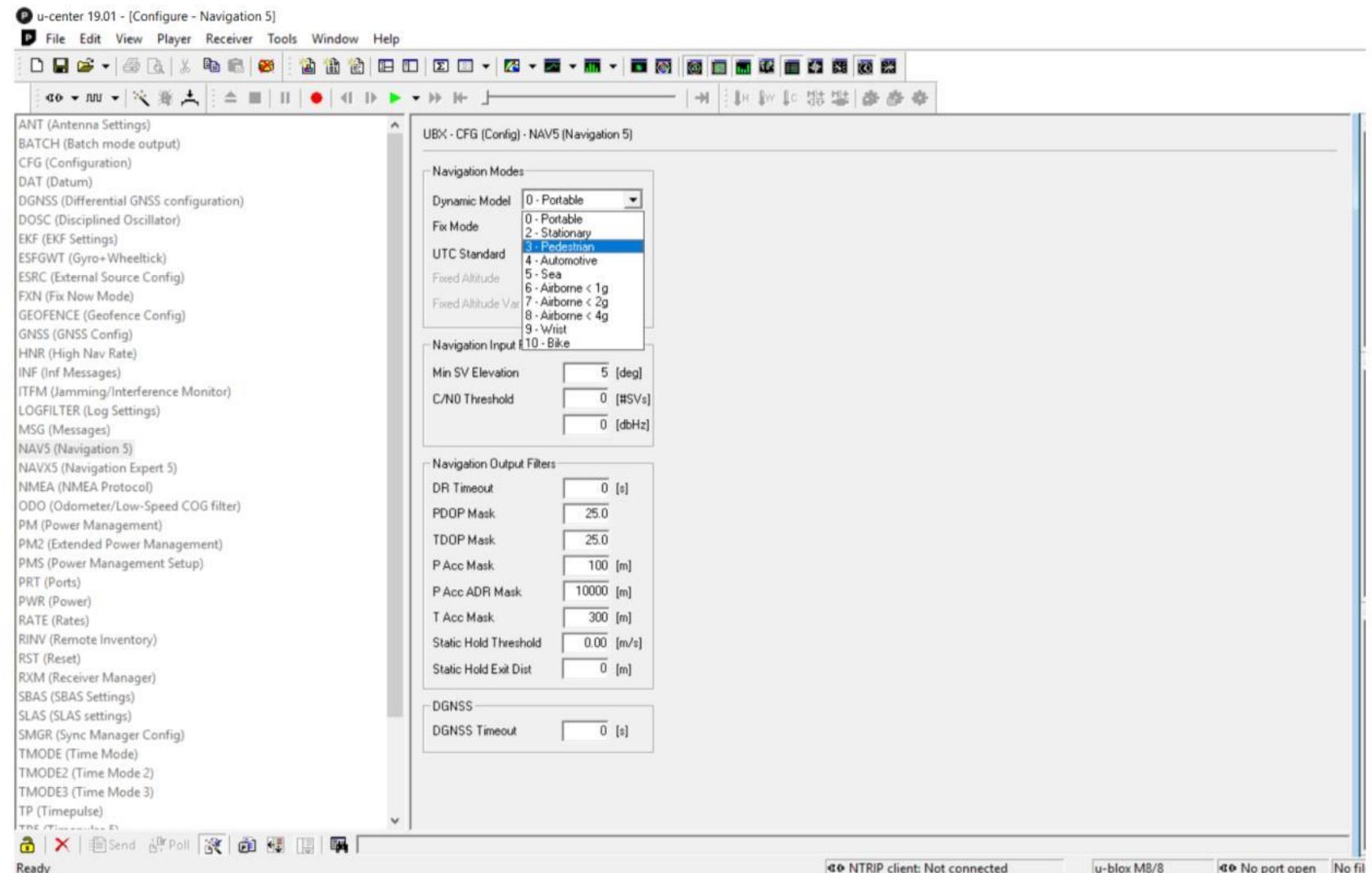
Nav5 (Navigation 5)

Dynamic Model

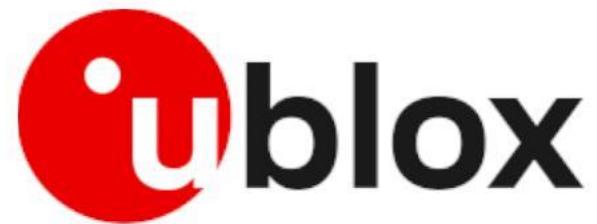
3. Pedestrian – better position hold

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

Hit Send icon to save

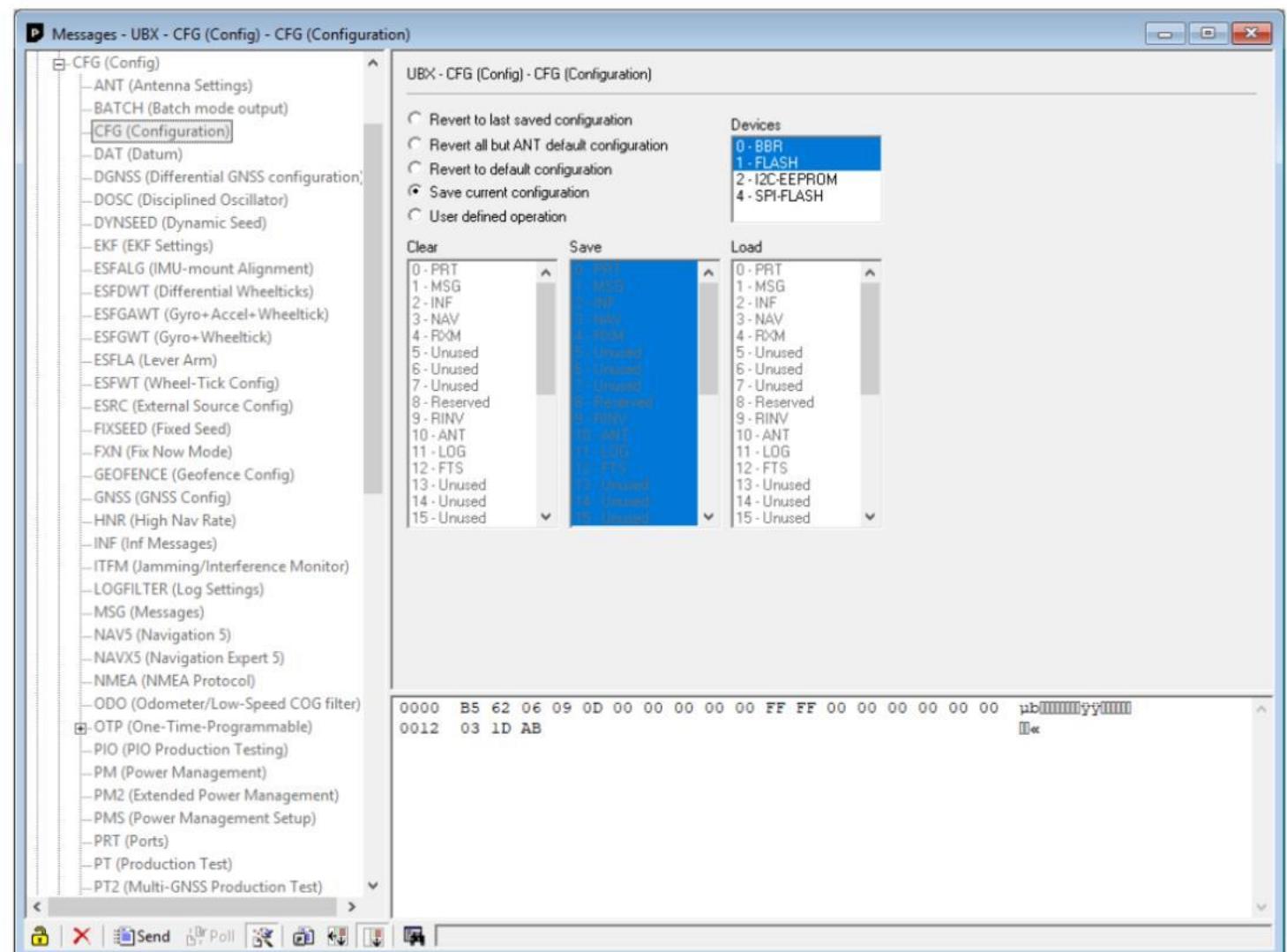


U CENTER



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

MultWii 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 lens

```
/*
 * 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

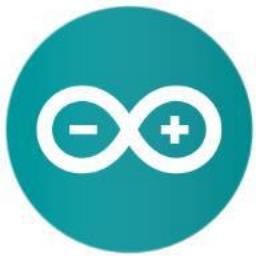
Done Saving.
```

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.h

```
// 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 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 chipsets based GPS with DIYDrones binary firmware (v1.6 or v1.8)
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 */

#define NMEA //for Ublox NMEA GPS - NMEA Protocol
#define UBLOX //for Beitian GPS - UBLX 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
```

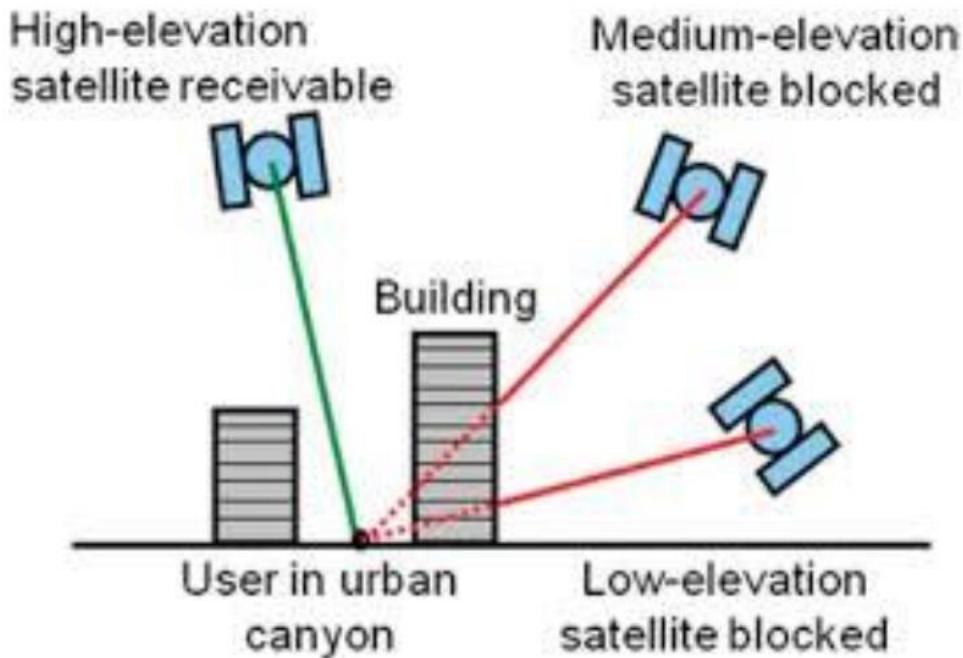
Arduino/Genuino Mega or Mega 2560, ATmega2560 (Mega 2560) on COM25
^ ENG 2:59 PM 30/09/2021

Depending on the GPS Version

Older formats uses NMEA protocol

Newer GPS uses UBLOX UBX Protocol

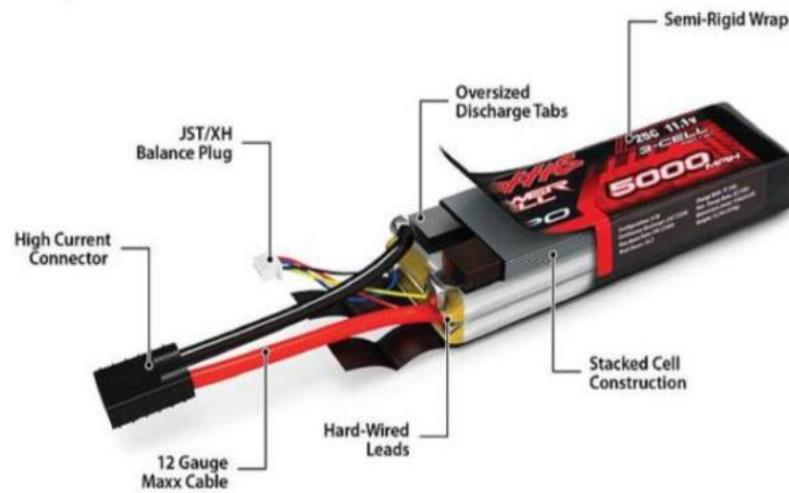
GPS



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

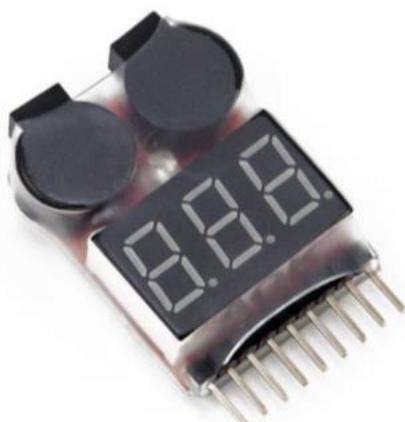
BATTERY STORAGE MODE IS 3.8V PER CELL

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



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 lens

```
/*
***** 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 BUZEER ! */

#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

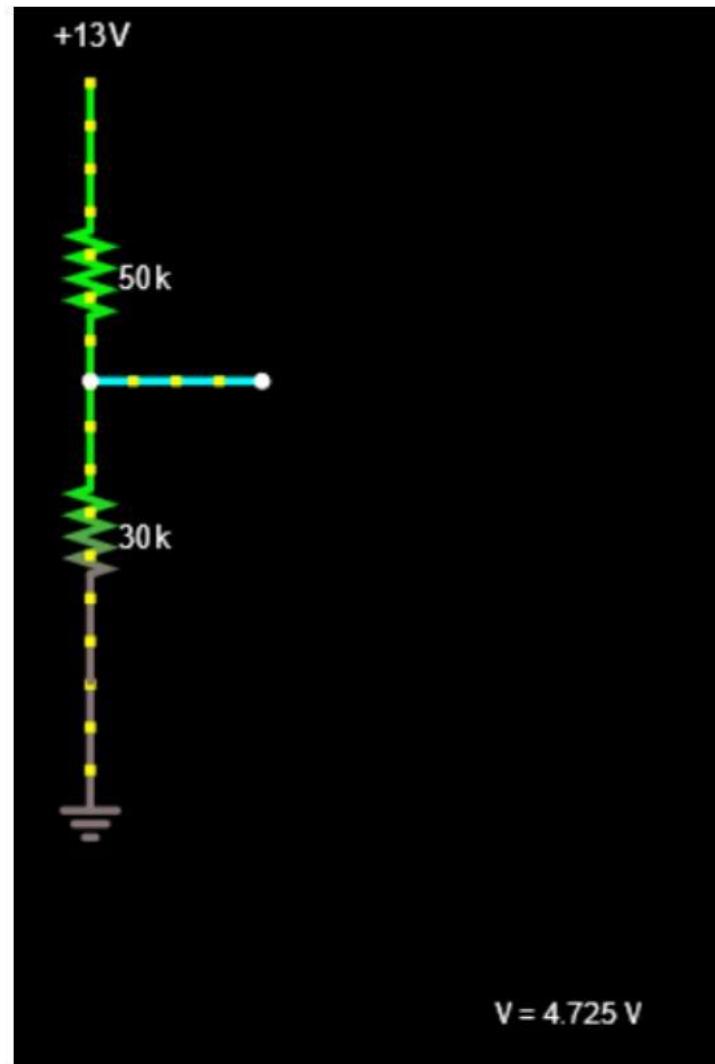
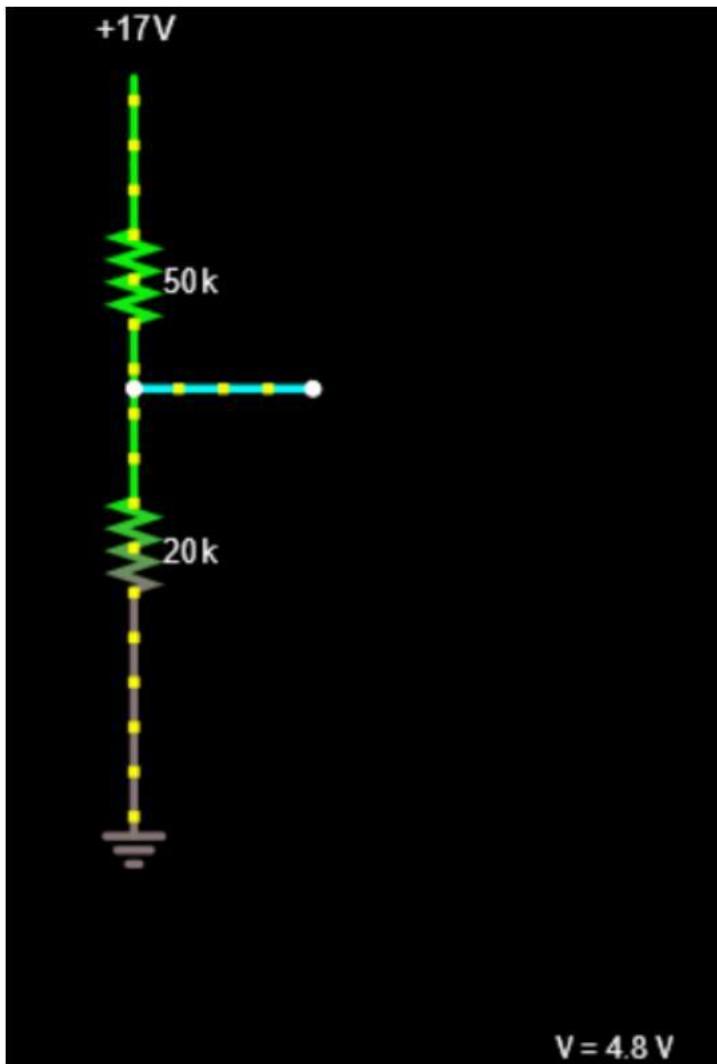
/* for V_BAT monitoring of individual cells
 * enable both VBAT and VBAT_CELLS
 */
#ifndef VBAT_CELLS
#define VBAT_CELLS_NUM 0 // set this to the number of cells you monitor via analog pins
#define VBAT_CELLS_PINS {A0, A1, A2, A3, A4, A5} // set this to the sequence of analog pins
#define VBAT_CELLS_OFFSETS {0, 50, 93, 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 ENG 13/02/2020

The bottom of the screen shows a Windows taskbar with several icons: Start, Search, File Explorer, Firefox, Paint, and others.

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 A0 ANALOG PIN
OF THE ARDUINO TO READ THE
BATTERY VOLTAGE

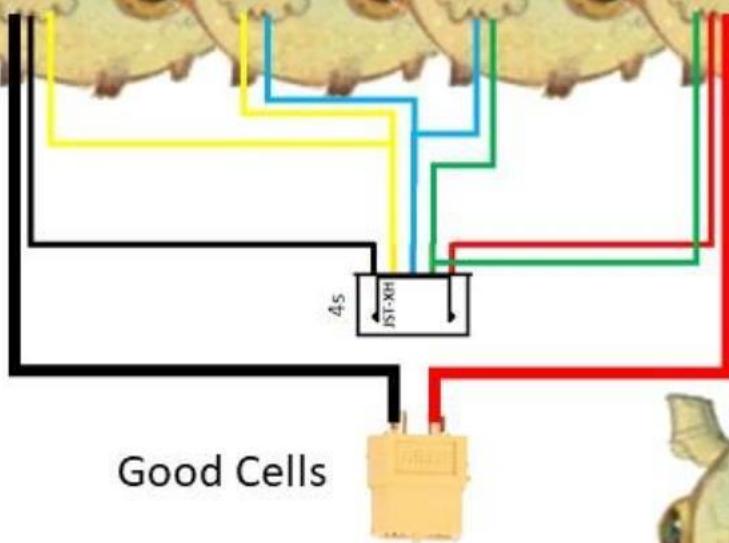
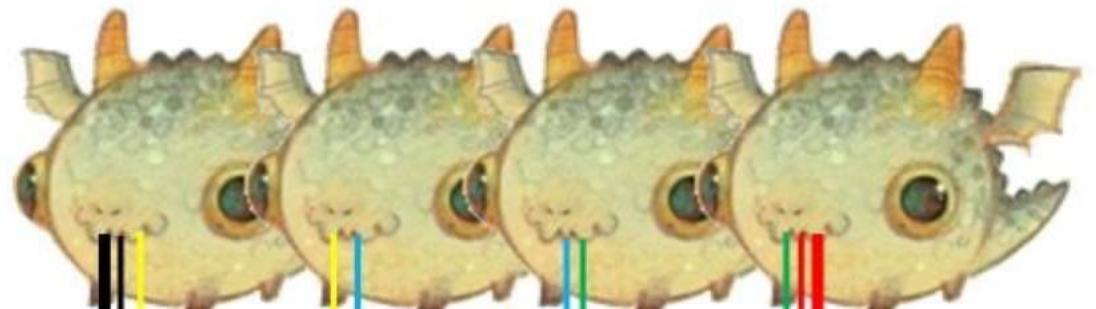
SWITCH THE VBAT JUMPER
ACCORDINGLY TO THE BATTER CELLS
YOUR USING
3S OR 4S

4.20

4.19

4.19

4.18



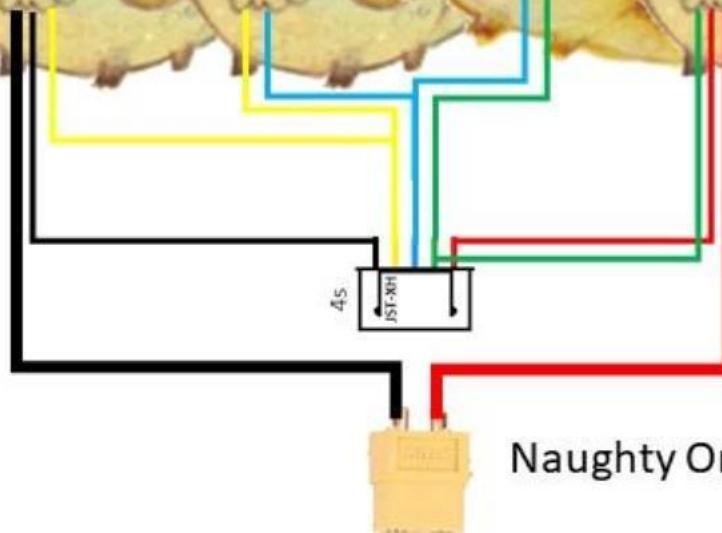
Good Cells

3.75

3.84

3.20

3.80



Naughty One

I always tell newbies to pretend they were
Orb Dragons

Treat them right and they will last a long time

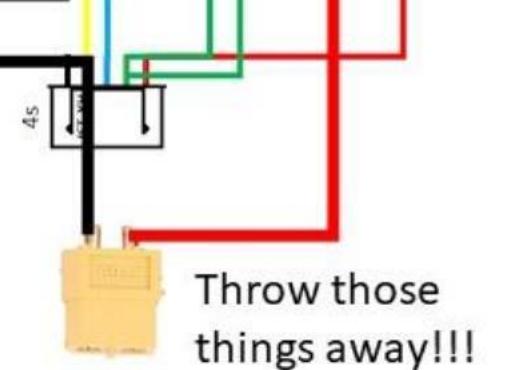
Abuse them and your asking for trouble

3.10

3.14

2.60

2.80



Throw those
things away!!!



BATTERY CHARGER



Standard Lipo Balance Charger
Nimh/Nicad/Lipo/Pb Mode 1s to 6s 5A

The Advantage of this is the ability to monitor your charge as well as resurrecting batteries that were over drain by 1V from drain



Cheap Balance Charger but no way users can monitor their battery condition

May post risk as you cannot monitor or emergency charge your battery

PIN ASSIGNMENTS

CONFIG.H



MultiWii - config.h | Arduino 1.8.2

File Edit Sketch Tools Help

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 ENG 21/02/2020

CAMERA STABILIZATION

CONFIG.H



Photo for illustration purpose only.

```
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 iem

/*
#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} // (*)
#define FORCE_SERVO_RATES {30,30,100,100,100,100,100} // 0 = normal, 1= reverse

***** [cam trigger] *****
/* 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 flapperons { Wing1, Wing2 }

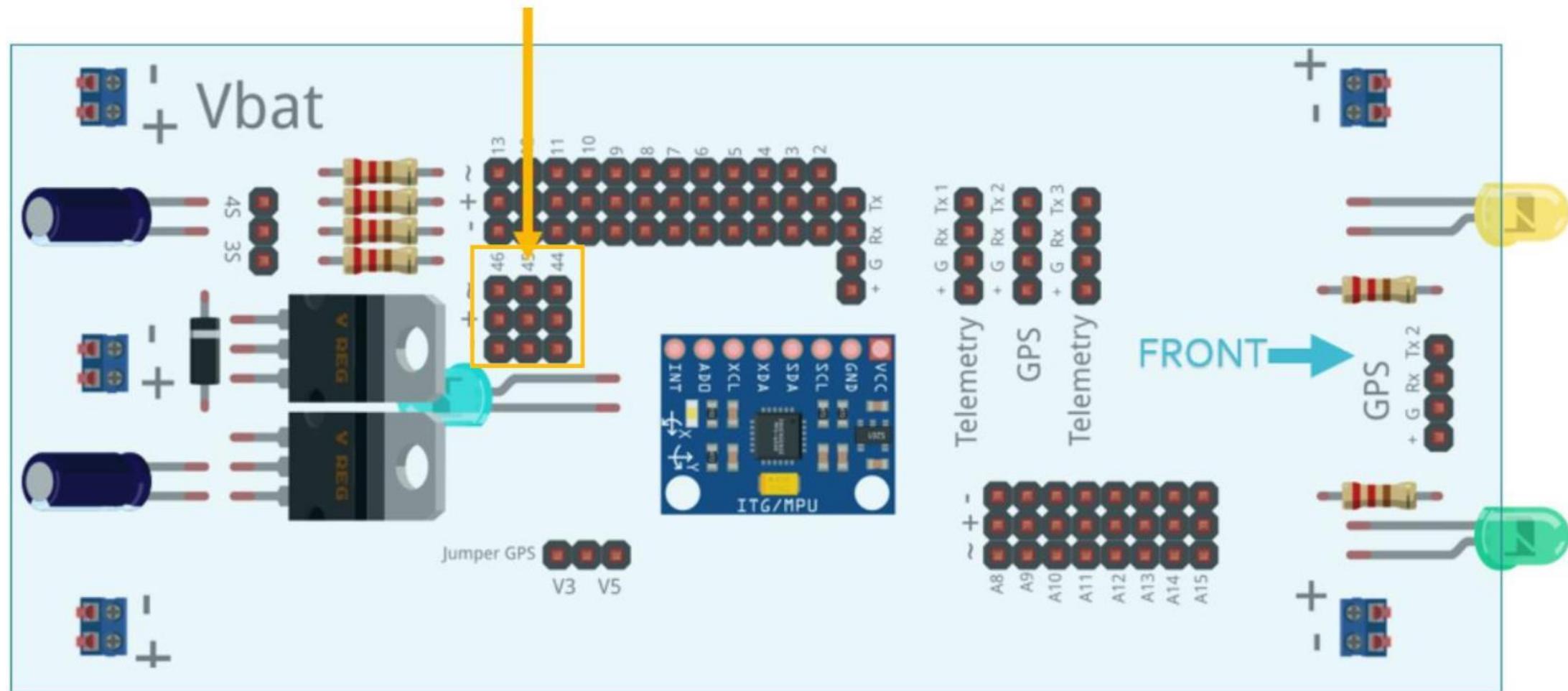
#define FLAPS // Traditional Flaps on SERVO3.
```

```
#define SERVO_MIX_TILT //D44 and D45 (120 Mix)
```

```
#define SERVO_TILT // D44 (Pitch), D45 (Roll)
```

```
#define CAMTRIG // Activate function D46 Trigger PWM
```

```
#define CAM_TIME_HIGH 1000 // HIGH state servo expressed in ms
```



Arm/DisArm

CONFIG.H



Arm/DisArm

Option for combination stick command to start and stop the drone

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

```
/* 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

/***************************** SERVOS *************************/
/* 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_MID {1500, 1500, 1500, 1500, 1500, 1500, 1500, 1500} // (*)
```

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

CONFIG.H



```
File Edit Sketch Tools Help
SynerduinoKwad Alarms.cpp Alarma.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

A Value on 200 will give a very distinct transfer */
#ifndef ACROTRAINER_MODE 200 // http://www.multiwii.com/forum/viewtopic.php?f=16&t=1944#p17437

***** Failsafe settings *****
/* Failsafe check pulses on four main control channels CH1-CH4. If the pulse is missing or below 985us (on any of these four channels) the failsafe procedure is initiated. After FAILSAFE_DELAY time from failsafe detection, the level mode is on (if ACC is available), PITCH, ROLL and YAW is centered and THROTTLE is set to FAILSAFE_THROTTLE value. You must set this value to descending about 1m/s or so for best results. This value is depended from your configuration, AUW and some other params. Next, after FAILSAFE_OFF_DELAY the copter is disarmed, and motors is stopped. If RC pulse coming back before reached FAILSAFE_OFF_DELAY time, after the small guard time the RC control is returned to normal. */
#define FAILSAFE // uncomment to activate the failsafe function
#define FAILSAFE_DELAY 10 // Guard time for failsafe activation after signal lost. 1 step = 0.1sec - 1sec in example
#define FAILSAFE_OFF_DELAY 200 // Time for Landing before motors stop in 0.1sec. 1 step = 0.1sec - 20sec in example
#define FAILSAFE_THROTTLE (MINTHROTTLE + 200) // (*) Throttle level used for landing - may be relative to MINTHROTTLE - as in this case

#define FAILSAFE_DETECT_THRESHOLD 985

***** DFRobot LED RING *****
/* I2C DFRobot LED RING communication */
#define LED_RING

***** LED FLASHER *****
#define LED_FLASHER
///#define LED_FLASHER_DDR DDRB
///#define LED_FLASHER_PORT PORTB
///#define LED_FLASHER_BIT PORTB4
///#define LED_FLASHER_INVERT
///#define LED_FLASHER_SEQUENCE 0b00000000 // leds OFF

Arduino/Genuine Uno on COM8
^ ENG 9:54 AM 22/07/2021
```

THROTTLE FAILSAFE –WHAT THE THROTTLE INPUT WOULD BE IF SIGNAL IS LOST BETWEEN THE SYNERDUINO BOARD , RECEIVER AND TRANSMITTER



FLYWIIGUIGR OUND STATION

Download the FlyWiiGUI groundstation and open FlywiiGUI.exe

Name	Date modified	Type	Size
210130-0301	30/04/2021 3:01 PM	File	3 KB
210814-0408	14/09/2021 4:08 PM	File	3 KB
212812-0428	12/06/2021 4:28 PM	File	3 KB
214012-0340	12/06/2021 3:40 PM	File	3 KB
AForge.Controls.dll	25/01/2015 1:15 PM	Application extens...	44 KB
AForge.dll	25/01/2015 1:15 PM	Application extens...	17 KB
AForge.Imaging.dll	25/01/2015 1:15 PM	Application extens...	248 KB
AForge.Math.dll	25/01/2015 1:15 PM	Application extens...	67 KB
AForge.Video.DirectShow.dll	25/01/2015 1:15 PM	Application extens...	52 KB
AForge.Video.dll	25/01/2015 1:15 PM	Application extens...	19 KB
AForge.Video.FFMPEG.dll	25/01/2015 1:15 PM	Application extens...	60 KB
avcodec-53.dll	25/01/2015 1:15 PM	Application extens...	13,181 KB
avdevice-53.dll	25/01/2015 1:15 PM	Application extens...	342 KB
avfilter-2.dll	25/01/2015 1:15 PM	Application extens...	870 KB
avformat-53.dll	25/01/2015 1:15 PM	Application extens...	2,405 KB
avutil-51.dll	25/01/2015 1:15 PM	Application extens...	135 KB
FlyWiiGUI.exe	30/10/2021 11:41 ...	Application	6,945 KB
FlyWiiGUI.exe.config	28/02/2017 5:31 PM	CONFIG File	1 KB
FlyWiiGUI.exe.manifest	30/10/2021 11:41 ...	MANIFEST File	30 KB

The FlyWii GUI is a free updated version of the MultiWii WinGUI. It serves as the ground control station for the MultiWii 2.4 controller software.

FlyWii GUI is currently only supported for Windows 7/8/10



[Download](#)

Latest Release

FlyWiiGUI Ground Station Software .EXE*

FlyWiiGUI20

[Download](#)



DEDICATED USER INTERFACE
AS CONFIGURATOR AND
GROUND STATION FOR THE
ARDUINO DRONE

FLYWII GUI



This screenshot shows the configuration and flight tuning section of the FlyWiiGUI. It features a large central area for setting various flight parameters. On the left, there are sections for 'Roll', 'Pitch', 'Yaw', 'Altitude', 'PosHold', 'PosHoldRate', 'Navigation Rate', 'Level', and 'Mag'. Each section contains sliders for 'P', 'I', 'D', and 'RC Expo' values. To the right of these sliders are two large black rectangular boxes labeled 'No Data'. On the far right, there's a detailed list of 'Navigation settings' with checkboxes for options like 'Enable GPS filtering', 'Enable GPS forward prediction filter', 'Don't reset home position at arm', etc. At the bottom, there are several numerical input fields for 'WP Radius (cm)', 'RTH Altitude (m)', 'Crosstrack gain', 'Max Nav speed (cm/s)', 'Min Nav speed (cm/s)', 'Max Nav banking (degree)', 'Land Speed', 'Safe WP distance (m)', 'Max Nav Altitude (m)', and 'Fence radius (m)'. A small icon of a remote control is visible on the left side of the bottom panel.



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)

PWM OUTPUT INDICATOR

PWM INPUT INDICATOR

CALIBRATION ACC / MAG

SAVE CONFIG

FLIGHT & GPS LOGS

ALTITUDE (BARO)

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

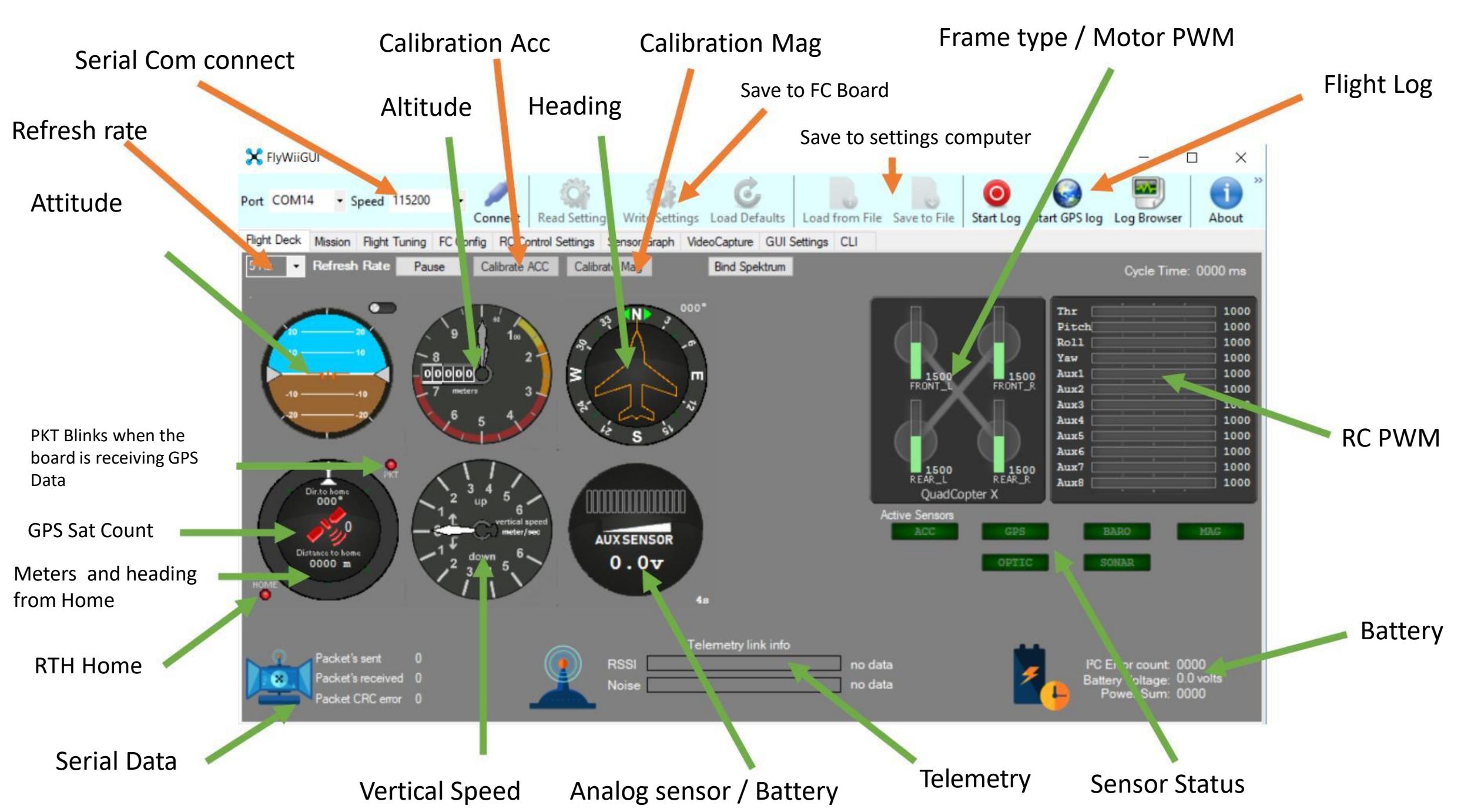
GPS SATELLITE COUNT
(4 SATS FOR 3D FIX – IDEAL 7 SATS)

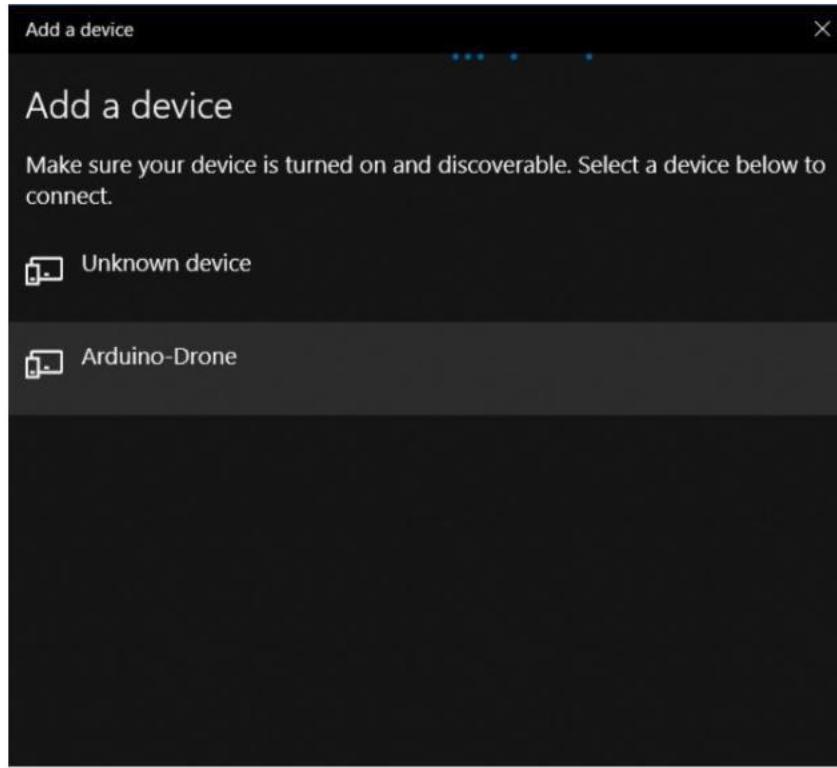
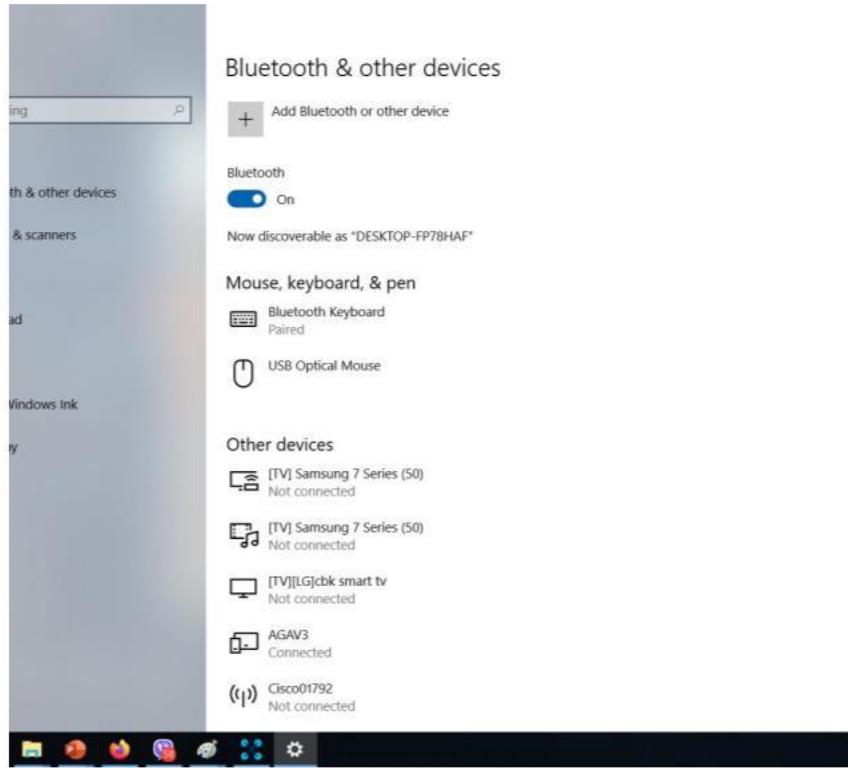
VERTICAL SPEED INDICATOR
(ACC Z AXIS)

PACKETS STATUS

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

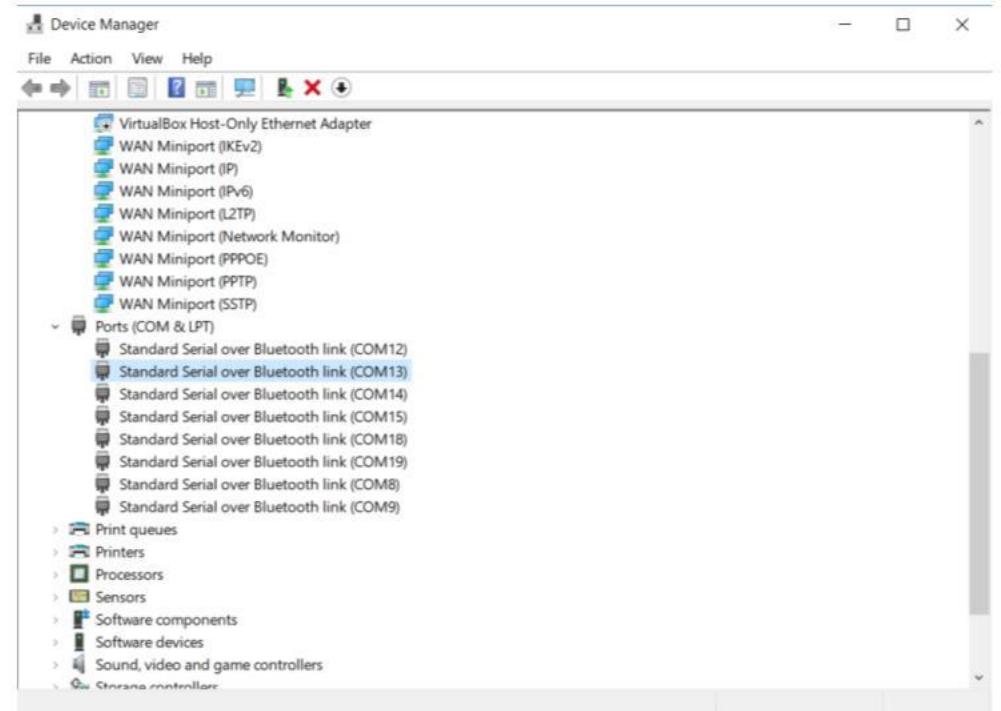
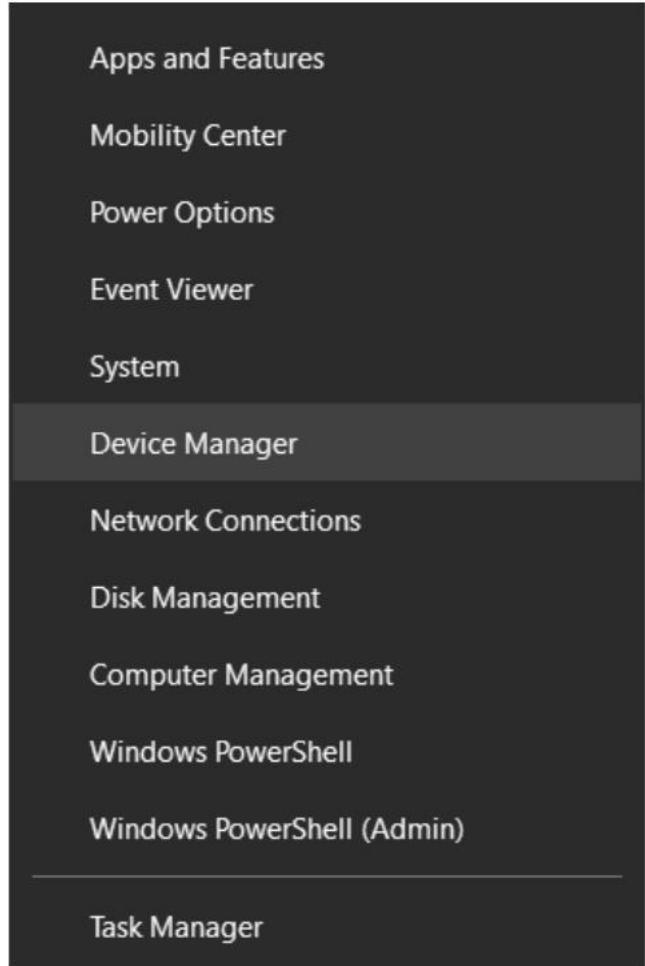






Adding Bluetooth on Windows Device Manager look for Arduino-Drone BT device

Take note on which Serial Com port its added to in Device Manager



in Device Manager Located in COM & LPT

Select the com port your Bluetooth is connected to .

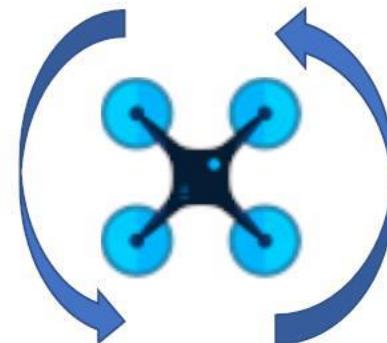
At this point Disconnect your Physical USB and your drone should be running on batteries using only the Bluetooth to communicate

Connect to the Drone with the associated COM port and Baud as found in your device manager





- Refresh Rate . Telemetry update speed
- Acc Calibration . Set the drone down on a level surface . Away from any metal objects for 30 secs.
- Mag Calibration . Move the drone 360 degrees in all axis within 1 min. while the blue Led flashes
- Mag Calibration must be perform when launching your drone in a new location for the first time. Pls verified the Compass if the drone heading matches your compass app in your phone.

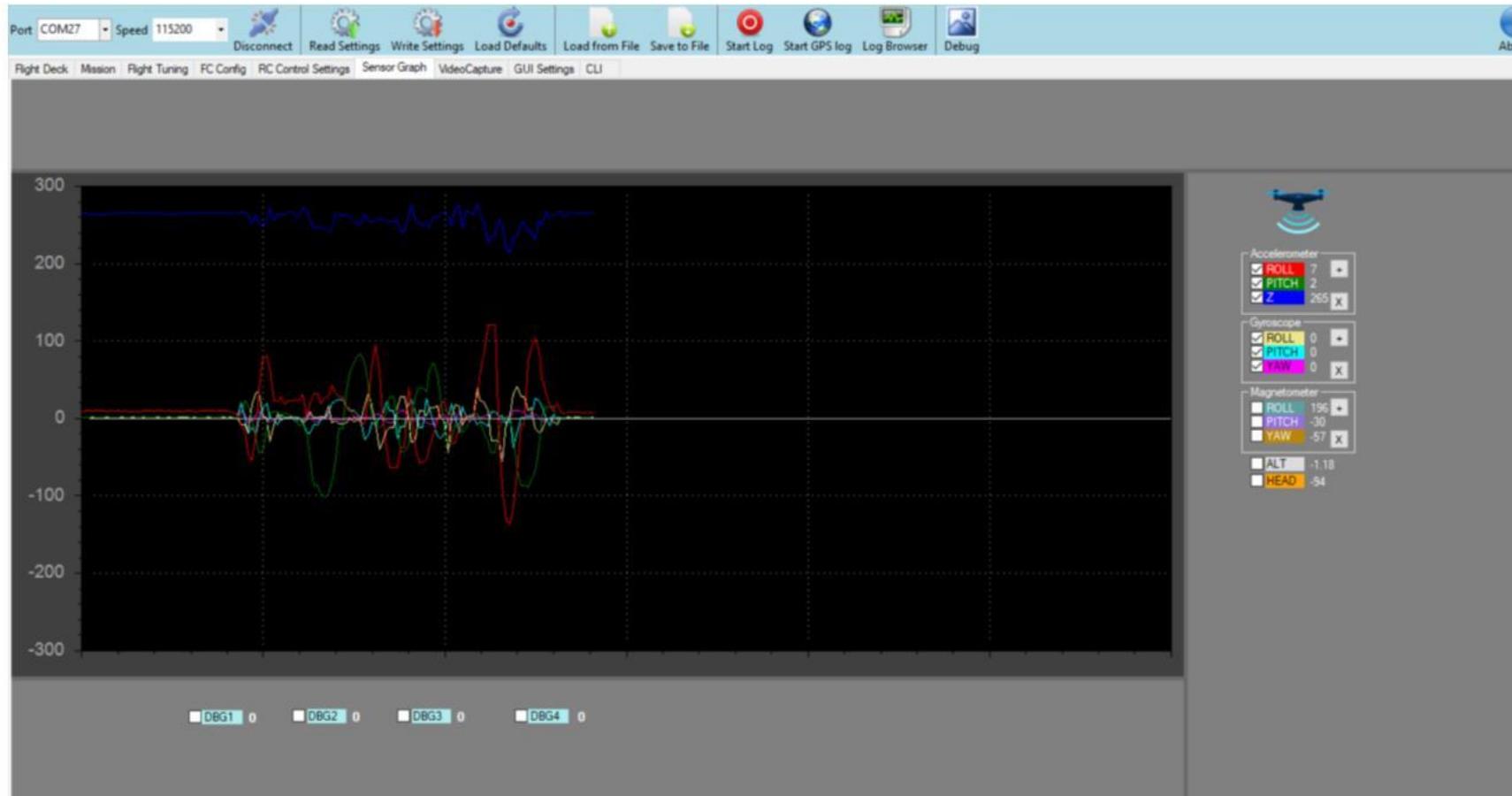


These Calibration must be perform after Parameter updates after Flashing the firmware
Blue LED would flash during these calibration processes



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



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 , and if you hold the drone upsidedown Z axis would show Negative numbers

the correct orientation
ACC

Roll Right + no#
Pitch nose down + No#
Z up + No#

GYRO

Roll Right + no#
Pitch nose down + No#
Yaw Right +No#

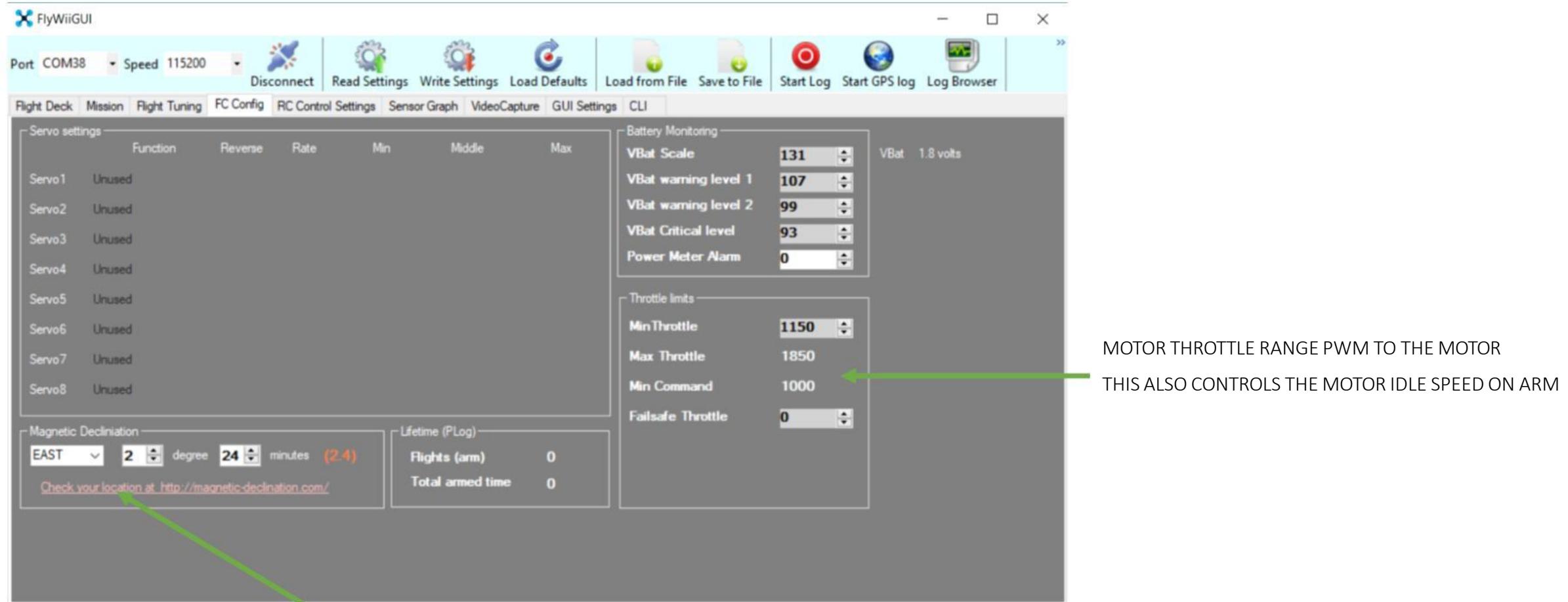
MAG

Mag & HEAD degrees
corresponds to the compass
(0 degrees = North)

BARO

Alt up +no#

FC Config Functions



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



BATTERY VOLTAGE CALIBRATION

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)

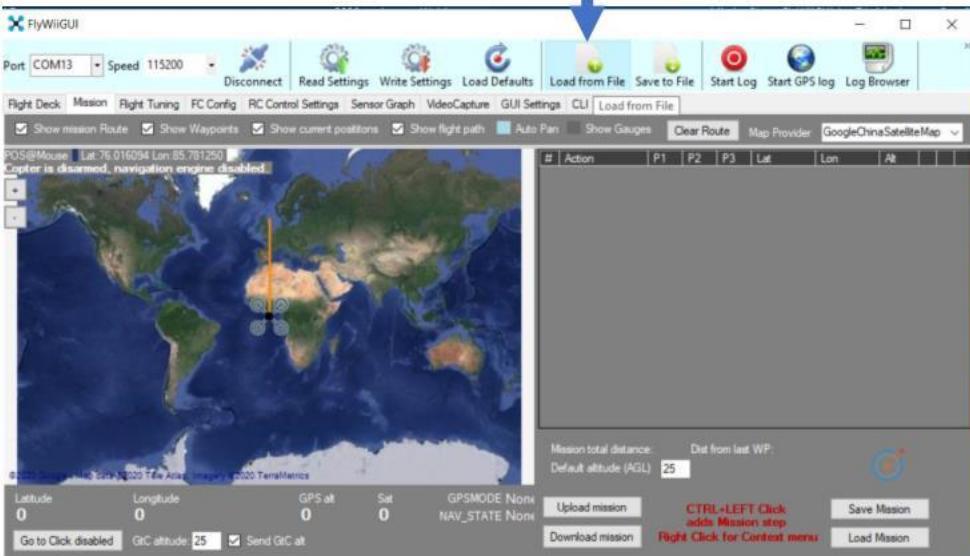
BATTERY CELL COUNT- ADJUST THIS DEPENDING ON THE NUMBER OF CELLS

THIS BOARD SUPPORTS 2S-4S BATTERY

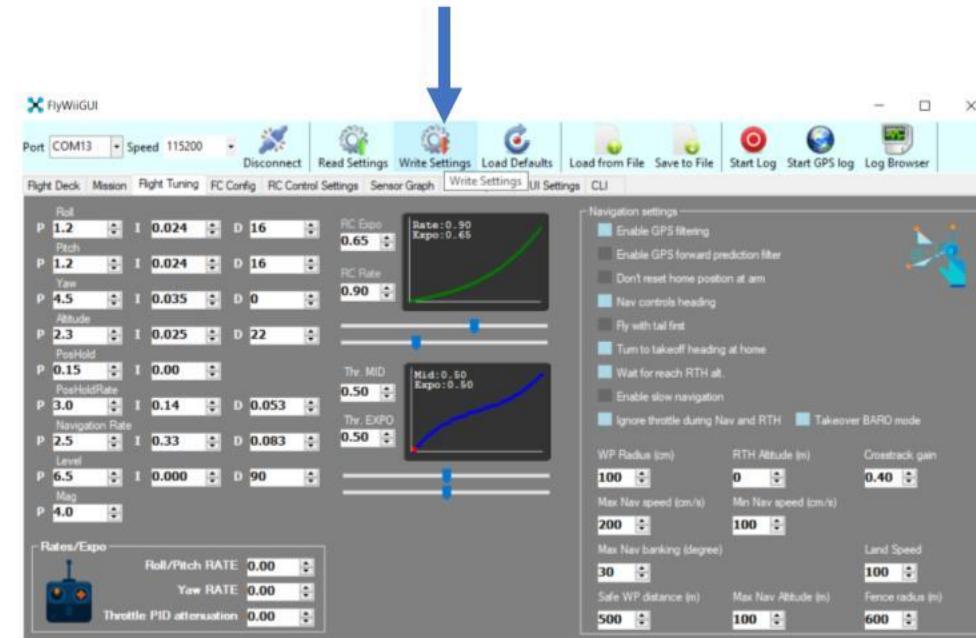
BATTERY

Download PID Parameters Preset Unzip and Load the PID file and Write settings after changes made in any of the parameters

Load the PID file



Write PID Parameter to drone



PID Parameters Preset*

PID 250mm v1.1 Slow Rate*

Download

250mm v1.1 Kwad - slow rate PID.mws

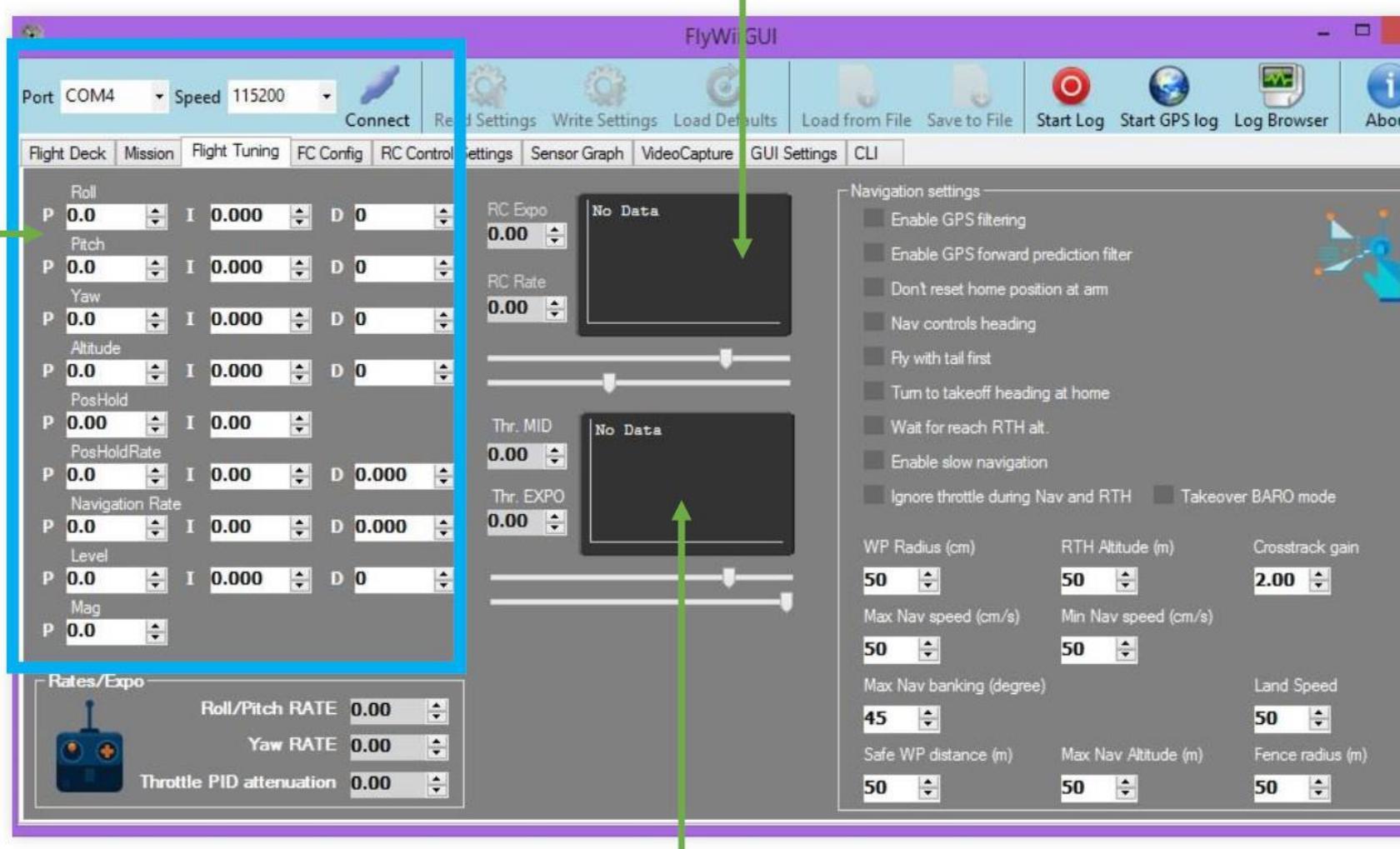


FLIGHT TUNING

FLIGHT TUNING
FOR STABILITY
AND CONTROL
(PID)

PROPORTION
INTEGRAL
DERIVATIVE

RC RADIO EXPO RATE CONTROL , THIS CONTROL
HOW RESPONSIVE YOUR DRONE RESPOND TO
YOUR STICK INPUT



THROTTLE CURVE EXPO THIS ALSO CONTROLS THE THROTTLE RESPONSIVENESS
AND THE DEAD ZONE FOR ALTITUDE HOLD

Understanding PID and its relation to stability and Controls

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
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 MultiRotor 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 destabilised.

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 MultiRotor 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 destabilised.

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

Aerobic flight: Requires a slightly higher P

Gentle smooth flight: Requires a slightly lower P

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 Multi-Rotor is returned to its original position.

A lower D will mean the Multi-Rotor will snap back to its initial position very quickly

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

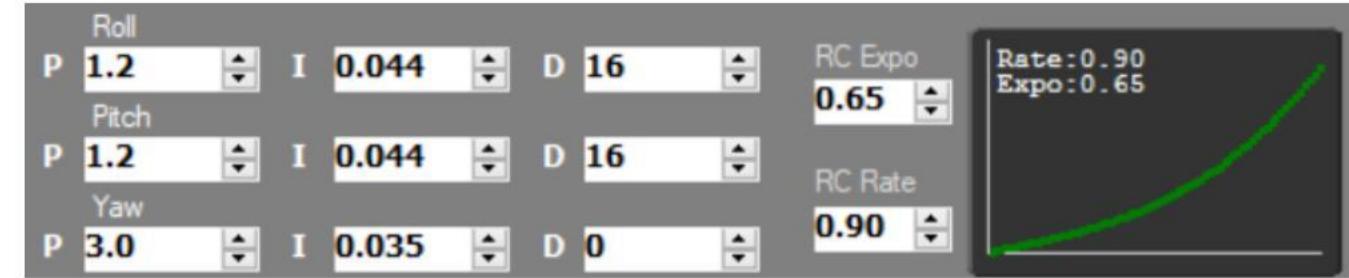
Decreasing value for D: Less dampening to changes. Reacts faster to changes

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 Multi-Rotor securely and safely in the air.
- 3 - Increase throttle to the hover point where it starts to feel light.
- 4 - Try to lean the Multi-Rotor 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 stabilisation you will feel it allow you to move over a period of time. That is OK
- 6 - Now try rocking the Multi-Rotor. 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.

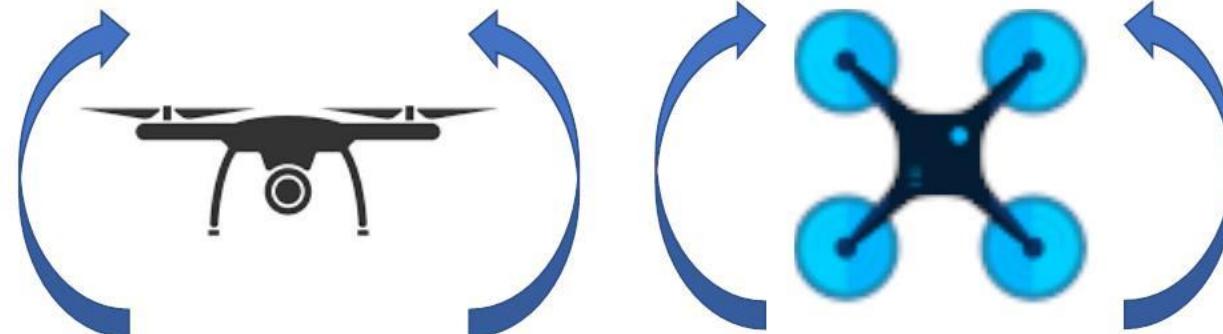


You will have to accept a compromise of optimal settings for stable hover and your typical mode of flying.

Obviously factor it towards your most common style.

Other factors affecting PID Taking known good PID values from an identical configuration will get you close, but bear in mind no two Multi-Rotors will have the same flying characteristics and the following items will have an impact on actual PID values:

- 1 - Frame weight /size / material / stiffness
- 2 - Motors - power / torque /momentum
- 3 - Position - Motor-->motor distance (I.E. frame size)
- 4 -ESC / TX - power curves
- 5 - Prop - diameter / pitch / material
- 6 - BALANCING
- 7 - Pilot skills





Advanced Tuning - practical implementation

For Aerobatic flying: Increase value for P until oscillations start, then back off slightly

Change value for I until wobble is unacceptable, then decrease slightly

Decrease value for D until recovery from dramatic control changes results in unacceptable recovery oscillations, then increase D slightly Repeat above steps

For stable flying (RC): Increase value for P until oscillations start, then back off quite a bit

Decrease value for I until it feels too loose /unstable then increase slightly
Increase value for D



PID : Level

Level								
P	8.0	▲	I	0.002	▲	D	80	▲

This will influence the flight characteristic with an accelerometer : this is Level Mode

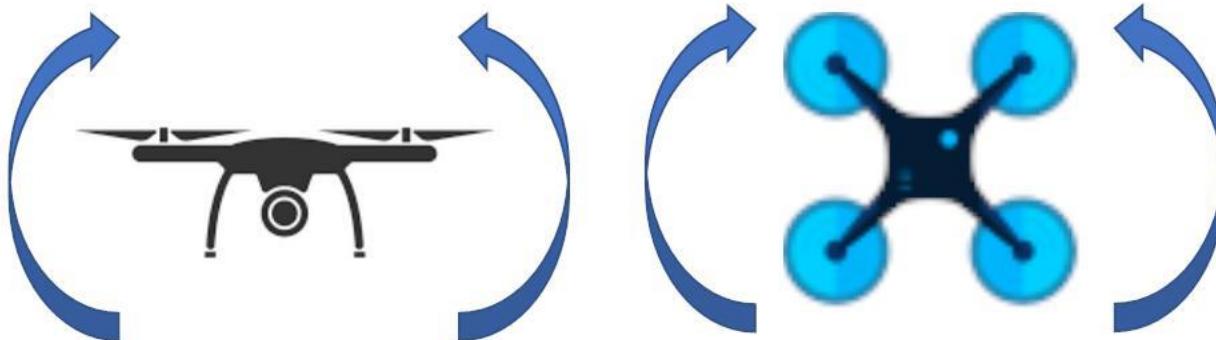
P is the dominant part of autolevel mode.

I will tell how much force must be applied when the measured angle error persists

D is used to clamp the maximum correction for autolevel mode

Increase value for P will make the autolevel mode stronger

For smooth operation the sum of P axis + P level should stay near the default value : if you decrease P for Roll and Pitch axis you can increase P Level





ALT Hold



The Barometer sensor is used to detect the altitude of your multirotor aircraft and is used for altitude hold mode. As the barometer sensor is not very precise and is quite noisy, detection of small up and down movements is impossible.

So small up and down movements are detected by the accelerometer Z axis.

Combination of these two sensors gives good altitude hold.

PID settings for ALT works like this:

P - Is how much the multirotor should rely on the barometer sensor. The higher the value is the stronger the multirotor relies on the Barometer reading.

I - Is used to compensate for drift caused by battery voltage drop during the flight. The higher the value is more the multirotor will react to voltage drops (or other varying factors over time).

D - Is how strong the multirotor should react to data from the accelerometer Z axis. It is used to react to small up and down movements that the barometer cannot accurately sense. The higher the value is the stronger the multirotor will react to small altitude changes.



1. So set the P and I to 0
2. Start to play with D value only. To high D may cause yoyo effect (up and down oscillations). With to low D copter will be not able to react strong/fast enough to hold altitude. Your goal here is to set D to the value when copter don't oscillate up and down and also holds altitude quite well for a not very long period of time. Copter will not hold altitude perfectly at this point during long periods. It will slowly drift up or down, but altitude should be quite stable in short periods.
3. Start to increase P to the point where copter holds altitude over long time period. If the value is to small the copter will drift slowly up and down. If the value is to high yoyo effect may appear. Goal here is to set it to the point where copter holds altitude for quite some time. Copter will still go slowly down due to battery voltage drop over time.
4. "I" is used to compensate the voltage drop. So start to increase the "I" value slowly until you get a perfect position hold during a very long time.
Now your altitude hold should be good enough.

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 multirotor 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 multirotor 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 multirotor 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.

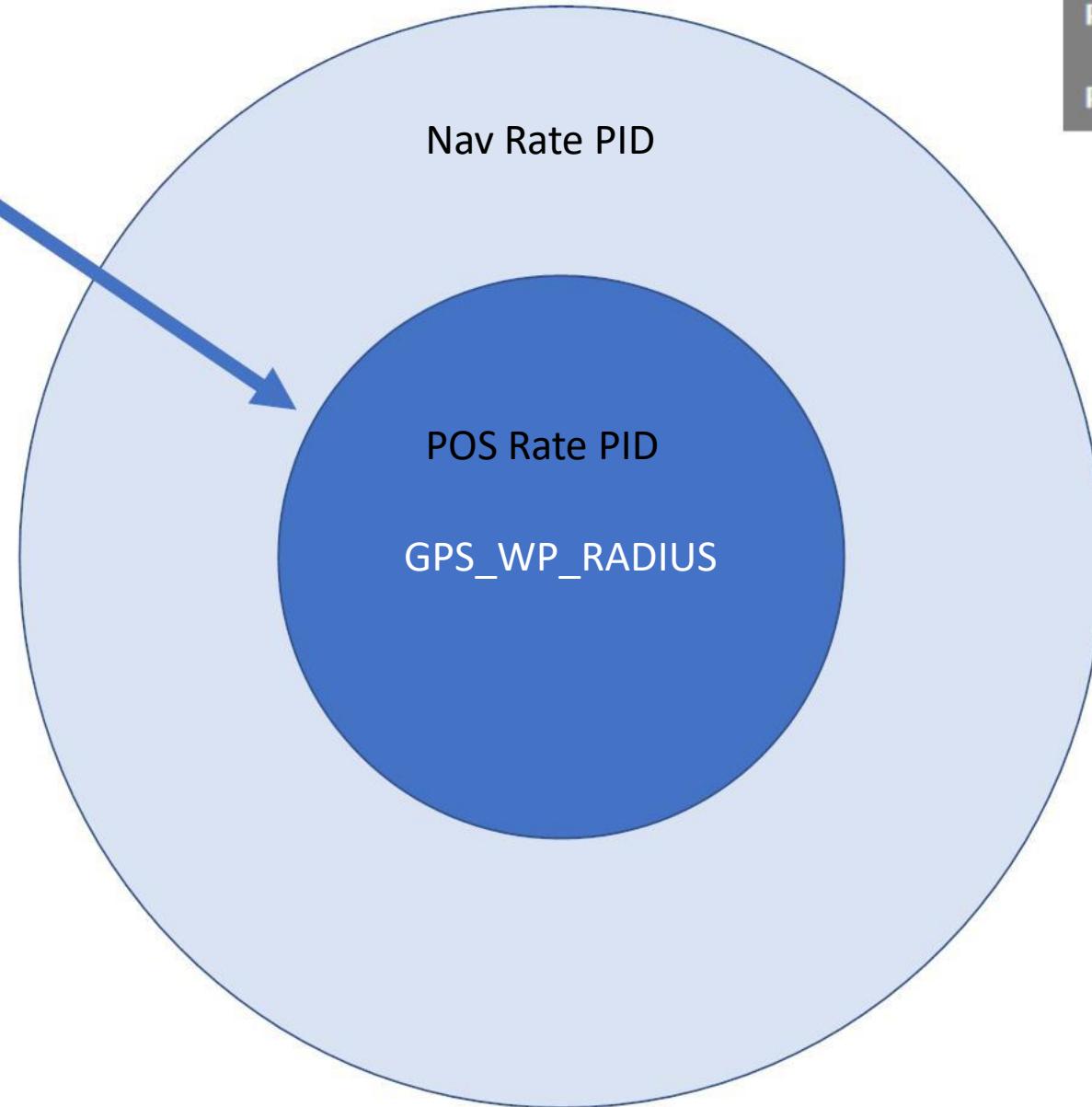
When the multirotor is within the defined distance of the hold location or waypoint (set by `GPS_WP_RADIUS` in config.h) the Pos Rate PID is used, when further away from the location the Nav Rate PID is used to return to within the defined waypoint radius.

To tune the Pos Rate PID, initially set P, I and D values to 0.

Gradually increase P until the multirotor begins to position hold with some drift.

Gradually increase D until the multirotor 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.



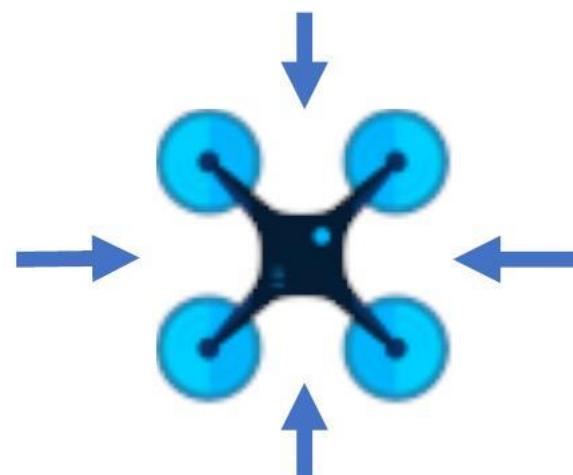
PosHold		P 0.15	I 0.00	D 0.053
PosHoldRate		P 3.4	I 0.14	D 0.083
Navigation Rate		P 2.5	I 0.33	D 0.083

To tune the Pos Rate PID, initially set P, I and D values to 0.

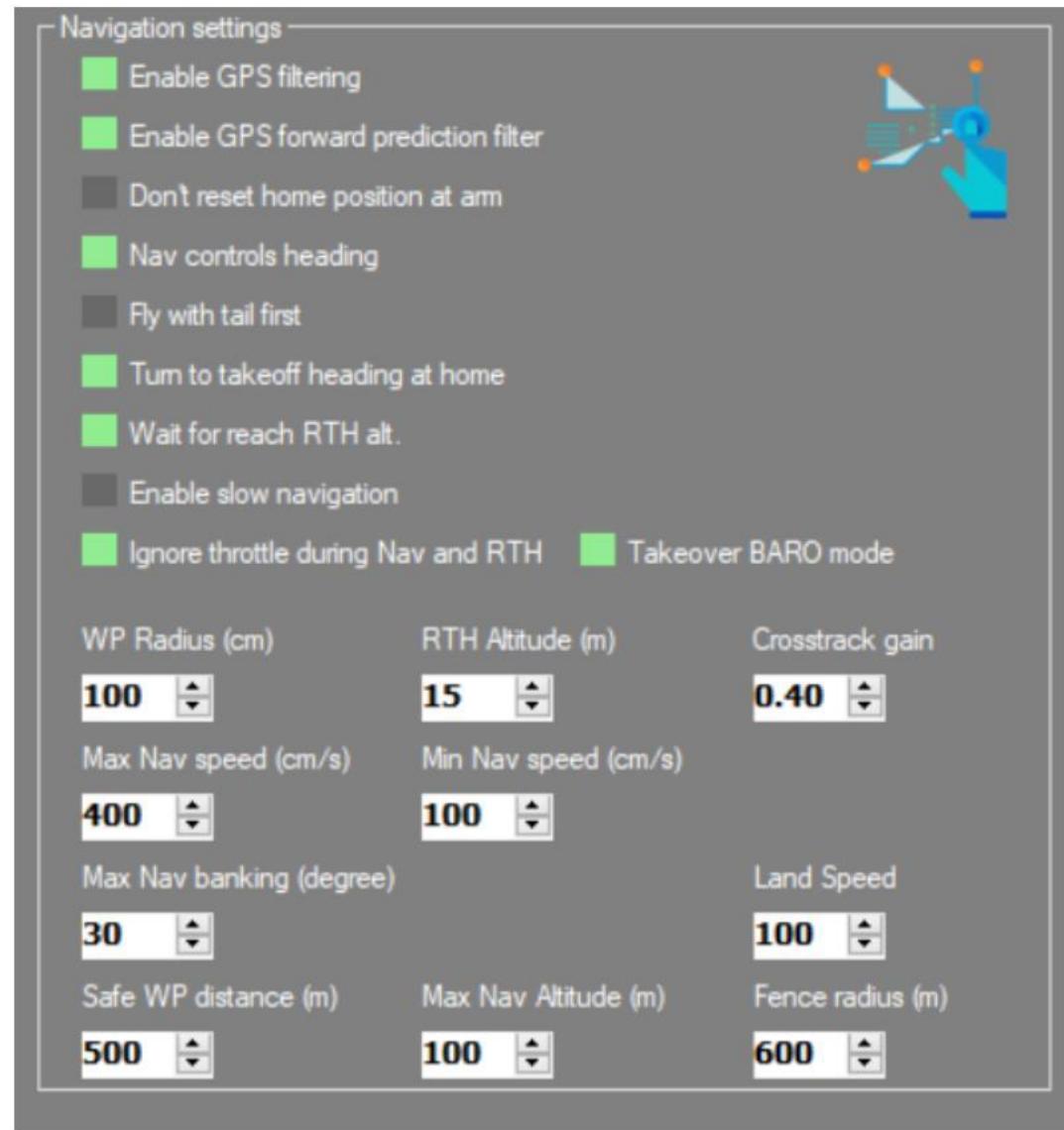
Gradually increase P until the multirotor begins to position hold with some drift.

Gradually increase D until the multirotor 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.



Other Navigation Functions



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

Max Nav Speed – Maximum speed the drone travels 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 travels when within the WP Radius

RTH Altitude – Altitude the drone will climb to when it's below the altitude in relation to its home point when the RTH is triggered set this to 0 to RTH at current altitude

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)

Max Nav Altitude – Max altitude the drone is capped to fly at

Land Speed – speed of descending for Landing cm/s

Safe WP Distance – max distance between waypoint before it's nullified

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 drone'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 drone

Nav Controls Heading – this points the drone to its next waypoint

Fly tail first – makes the drone fly reverse (don't use unless it's a camera pull out shot)

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

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

Ignore throttle and Take over Baro – as the name suggest disable throttle stick command from the controller when the drone is on mission mode

FLIGHT TUNING

FlyWiiGUI

The screenshot shows the FlyWiiGUI software interface with the 'Flight Tuning' tab selected. On the left, there's a list of flight controller parameters with their current values:

Mode	P	I	D
Roll	1.2	0.024	16
Pitch	1.2	0.024	16
Yaw	4.5	0.035	0
Altitude	2.8	0.010	15
PosHold	0.15	0.00	
PosHoldRate	3.0	0.14	0.053
Navigation Rate	3.0	0.33	0.083
Level	6.5	0.000	90
Mag	4.0		

Below these are two curve editors:

- RC Expo:** Shows a green curve with 'Rate: 0.90' and 'Expo: 0.65'. A green arrow points from the text 'RC RADIO EXPO RATE CONTROL , THIS CONTROL HOW RESPONSIVE YOUR DRONE RESPOND TO YOUR STICK INPUT' to this section.
- Throttle:** Shows a blue curve with 'Mid: 0.50' and 'Expo: 0.50'. A green arrow points from the text 'HIGHER NUMBER MEANS MORE RESPONSIVENESS THE DRONE TO STICK INPUTS' to this section.

At the bottom left is a 'Rates/Expo' section with a joystick icon:

Roll/Pitch RATE	0.00
Yaw RATE	0.00
Throttle PID attenuation	0.00

RC RADIO EXPO RATE CONTROL , THIS CONTROL HOW RESPONSIVE YOUR DRONE RESPOND TO YOUR STICK INPUT

HIGHER NUMBER MEANS MORE RESPONSIVENESS THE DRONE TO STICK INPUTS

FOR BEGINNERS WE RECOMMEND RC RATE AT .50-.60 THIS OFFER A MORE SLUGGISH RESPONDS OF THE DRONE AND NOT ADVICE ABLE FOR STRONG WINDS CONDITION

THROTTLE CURVE EXPO THIS ALSO CONTROLS THE THROTTLE RESPONSIVENESS AND THE DEAD ZONE FOR ALTITUDE HOLD

ADJUST THIS FOR YOUR THROTTLE RESPOND RATE AND WHERE YOUR CENTER STICK DEADBAND FOR ALTITUDE HOLD IS



RC Control Settings

WiiGUI

Port COM27 Speed 115200

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

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

AUX1 AUX2 AUX3 AUX4

	L	M	H
ARM			
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

For beginners we recommend the Horizon is Permanently active

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

BARO – Altitude Hold
MAG – Heading Hold
HEADFREE – Course Lock regardless of orientation

GPS Home – Return to Home (copter will climb to RTH altitude then Fly to Launch point)

GPS Hold – Hold Position ,

MISSION – fly a mission save from the mission tab

Thr 1500
Pitch 1500
Roll 1500
Yaw 1500
Aux1 1500
Aux2 1500
Aux3 1500
Aux4 1500

Live RC data



Missions

Note: Only functional for Mega 2560 Boards with GPS

Waypoint – the drone will travel between those points

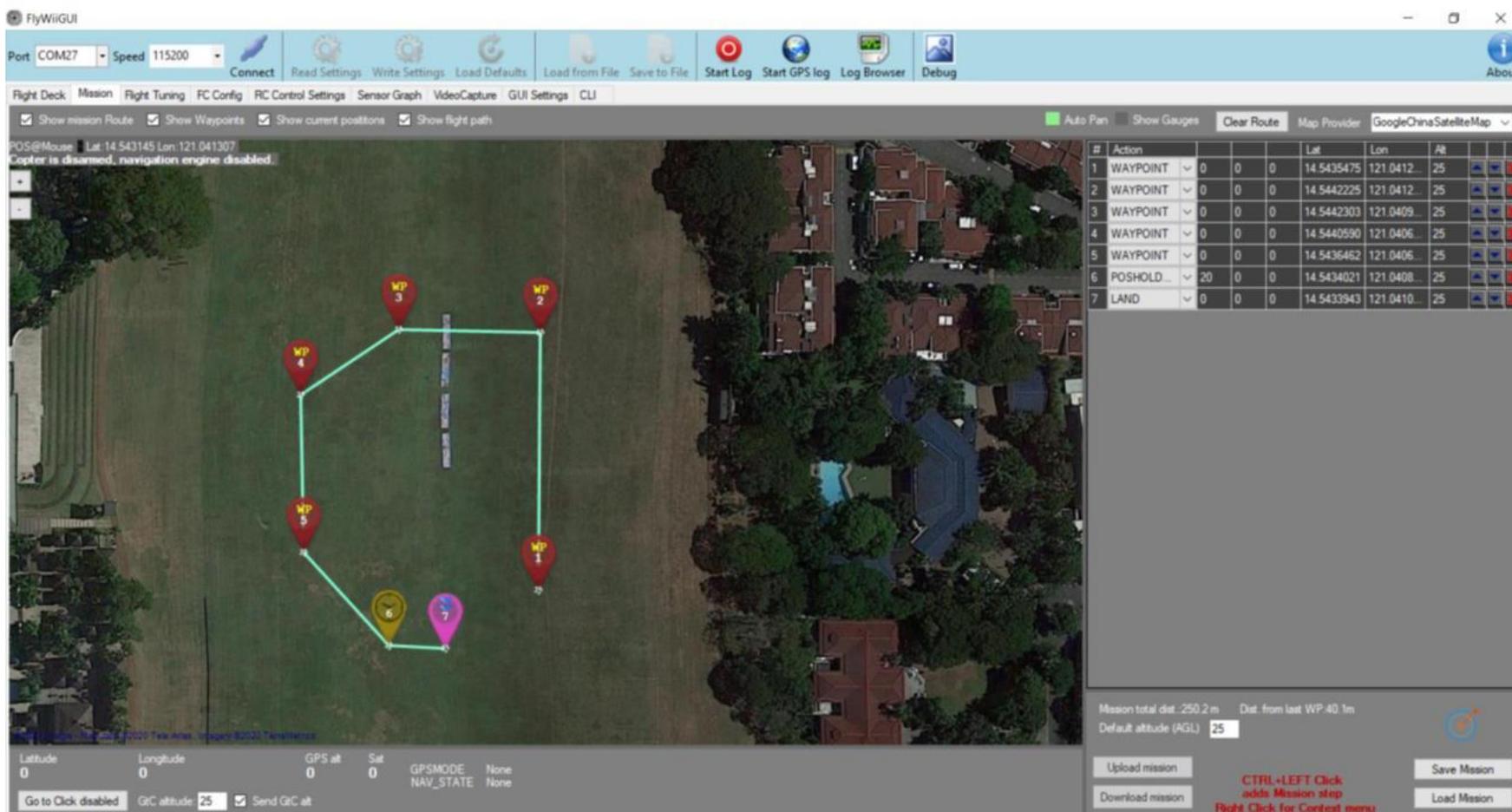
Time PosHold – Drone will wait X number of 00:00:00 then move to the next waypoint

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

Land – the drone will land once it reaches this point (**Must be placed at the end of the mission**)

RTH – the Drone will fly back to home position (**Must be placed at the end of the mission**)

Default Alt – Altitude in meters (for first Mission test waypoint with altitude 2m-3m Above Ground Level) And set missions with 2m-3m altitude with Nav speed of 100cm/s .



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 .

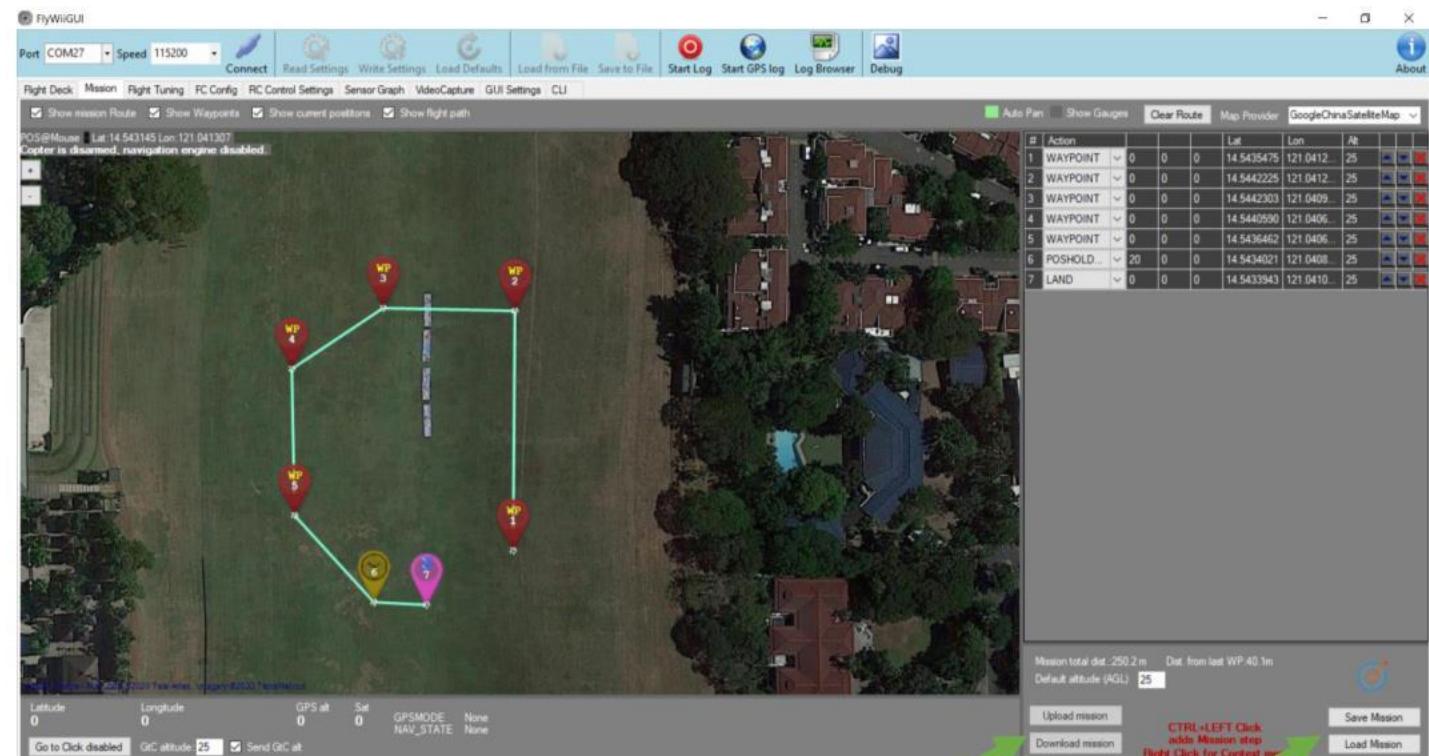
Any time you can switch out of it on hold or stabilize mode



Missions

Prerequisite and process for a good mission , Points to test before performing a mission

1. Drone is flying stable in horizon and Alt hold mode , holding altitude consistently less than 1m variation over 1 minute period . Tune PID and altitude PID when necessary.
2. Drone is flying stable and holding position in GPSHold mode and Alt Mode not deviating with in a 1 x1 Meter Imaginary box , tune PosHold Rate PID when necessary
3. RTH – set RTH Altitude to 0 , Max Nav speed to 100cm/s , set aux switch to RTH ,Baro , Mag and write settings ,Fly the drone 5 Meters away from the Launch site and activate the RTH Aux switch ,see if the drone returns back to home position and holds position when arrive . Tune Navigation Rate PID when necessary



Mission upload to /download from Drone

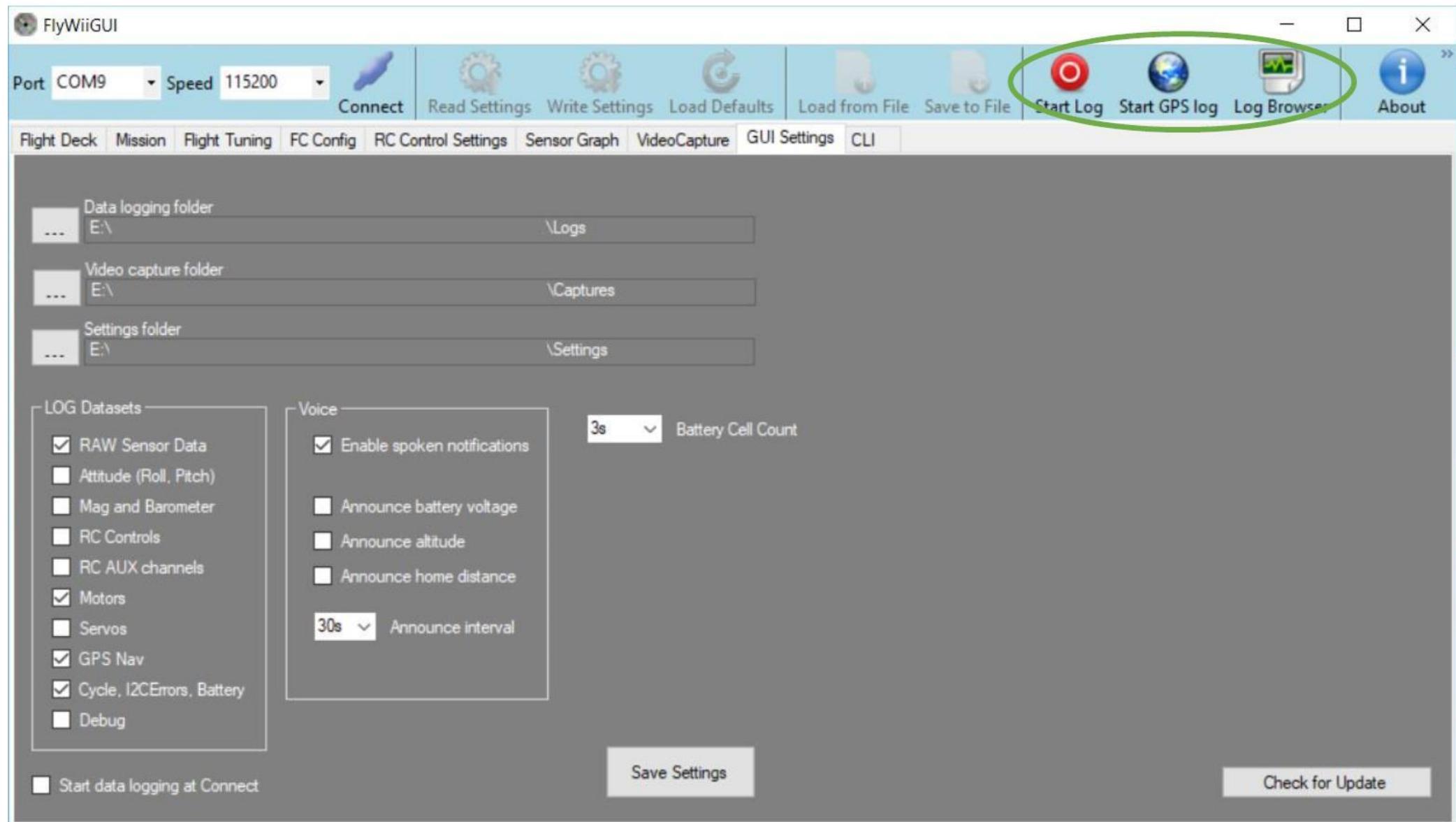
Mission Save to /Open from File



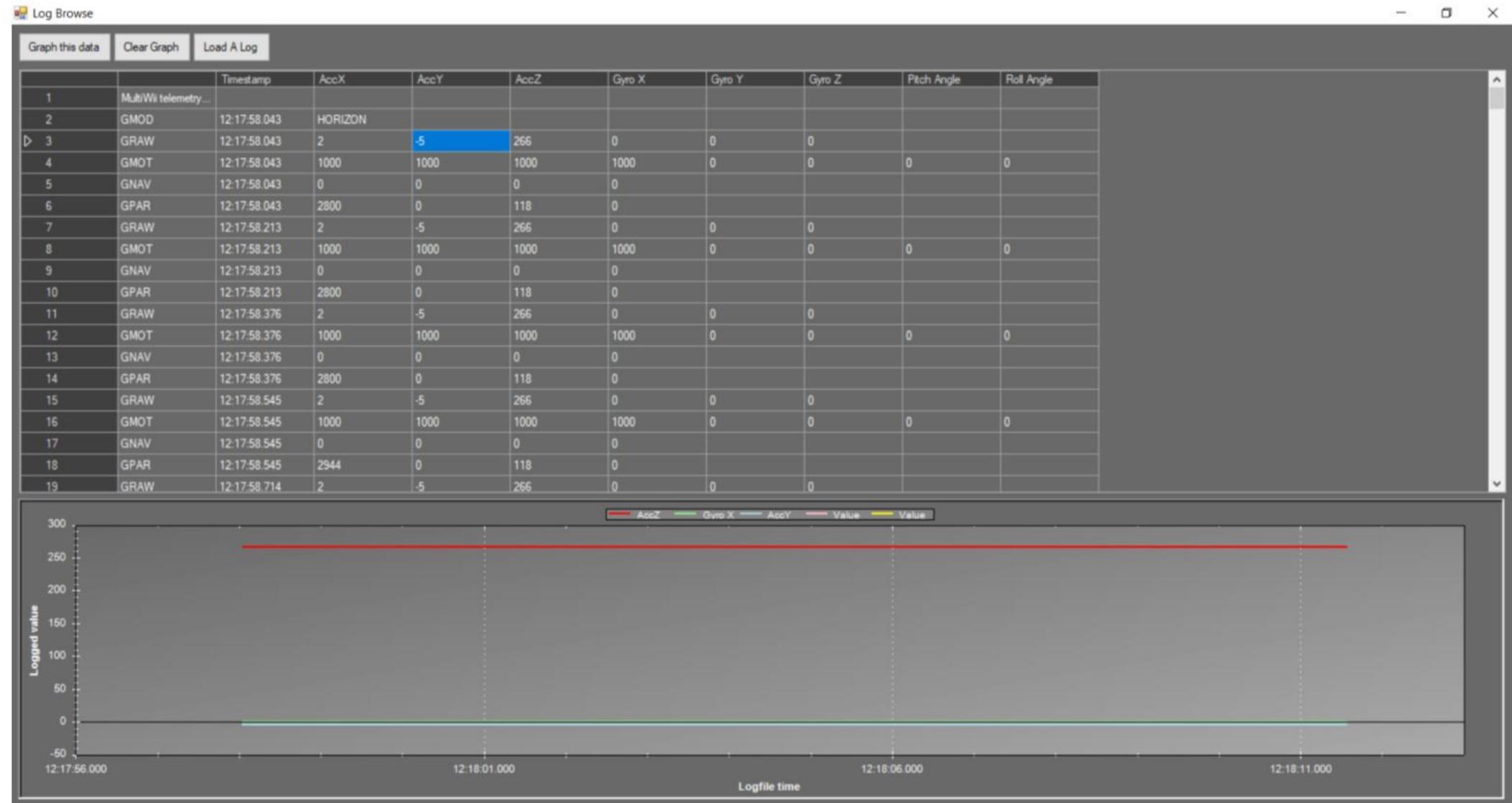
Data Logging

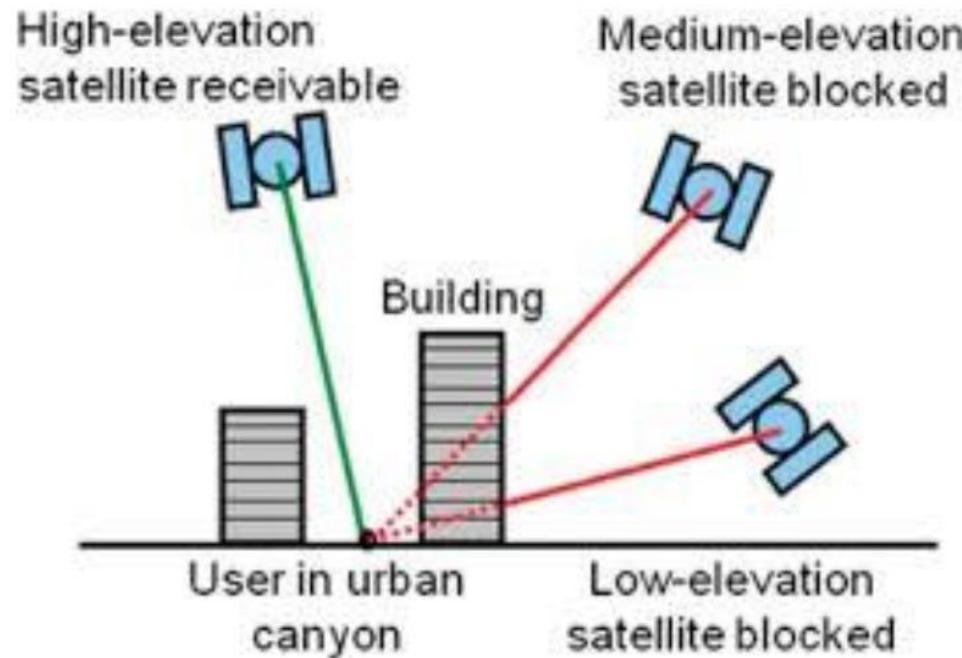


GUI Settings (where you save your PID ,Flight Logs and Video Logs)



Log Browser



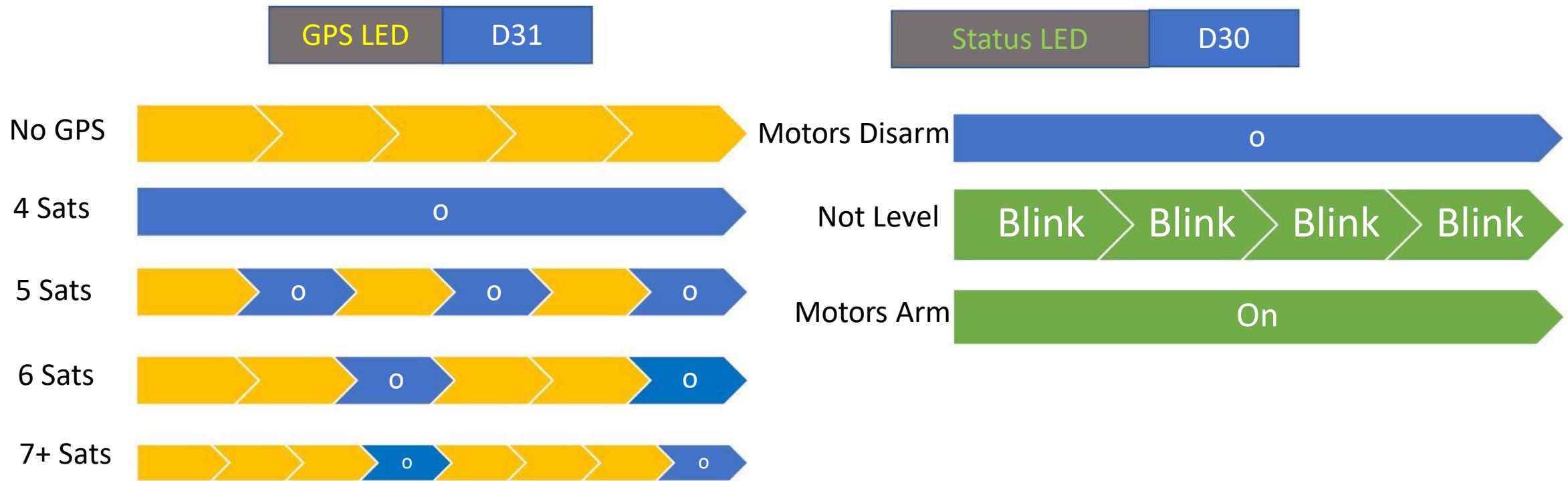


Note : GPS require a clear open area to get a proper fix and accuracy minimum 7 satellites but 10+ are Ideal

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

LED Indicator

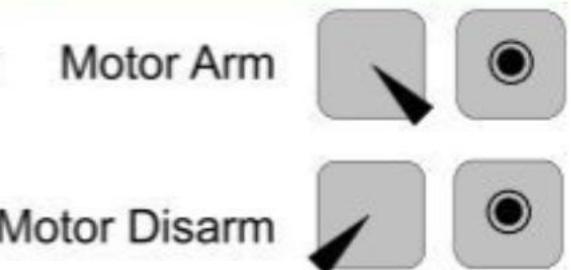


indicate a valid GPS fix by flashing the LED

- 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 +

And your much Done on your setup

For Mode 2 Hold 2 seconds



Cannot Arm Motors

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

Tests motors with Props off

Baro and Mag preferably switch off when Arming

Pls calibrate ACC and Mag in the Dashboard



Before your First Flight one Final procedure known as pre flight and Tuning tests



Before your First Flight one Final procedure
known as pre flight and Tuning tests

Note: You may need to build a suspended Test Rig
to test your drone's

RC mapping
EXPO rates
PID Tuning
Motor Rotation & Propeller installation

