# CleanFlight and BLHeli_32 Configuration
# Aerial Robotics Lab

Ali Grivani

August 27, 2019

# Contents

# 1 Introduction

This document is written to have a record of the aerial robotics' lab activities. These notes mainly cover laboratory equipment set up, mechanical, electrical or software issues.

# 2 Software Configurations

## 2.1 Connecting ESC to BLHeliSuite through flight controller

The procedure is straightforward as follows[1]:

1. Make sure the drivers are installed on your system and they are up-to-date.
2. Download the latest BLHeliSuite configurator for BLHeli32 here.
3. Connect the ESC to the flight controller. Connect the motors to the ESC (ESCs) and plug-in the battery.
4. Choose the correct COM Port and press Connect button.

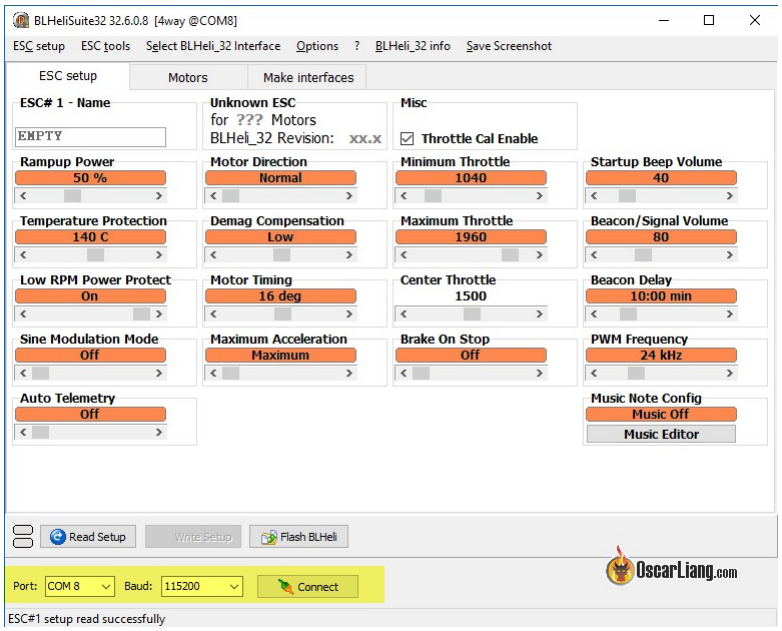


Figure 1: Connecting BLHeliSuite32

5. Click on "Read Setup" or "Check". You should see the following changes in the BLHeli interface.

[1]Some images are obtained from Oscar Liang's website

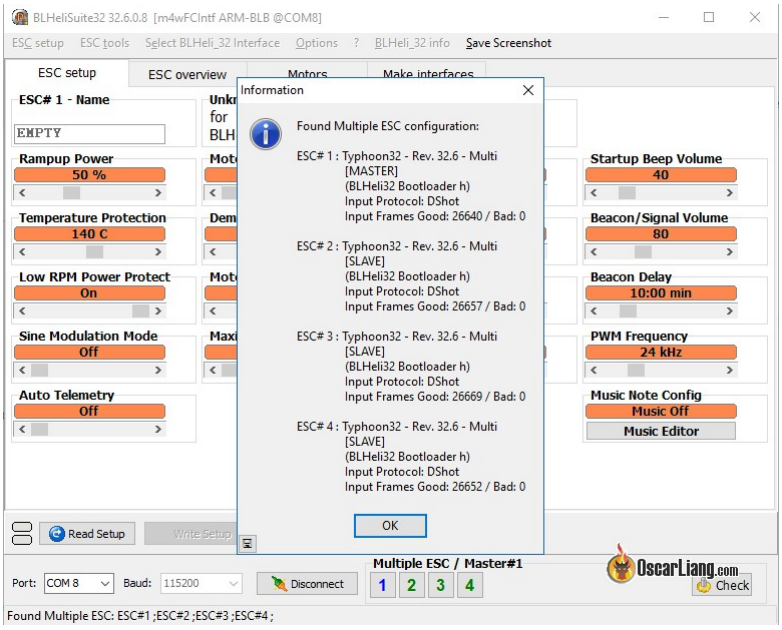


Figure 2: Multiple ESC Configuration Should be Found

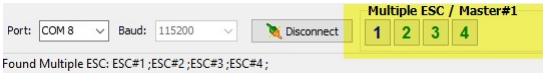


Figure 3: Multiple ESCs are found and they are shown in the boxes

If any of the motors (ESCs) were not recognized by the software, read section (**??**).

6. The first square for ESC #1 is blue meaning it is the master ESC and the rest are called the slaves. If you write any settings to the master ESC, the slaves will receive the same configuration. The slave ESCs are shown with green font.

## 2.2 ESC calibration

Usually, when ESCs are first used, they must be calibrated. The reason is explained in the section (**??**). In this section we are going through the steps of calibration using BLHeliSuite32 software. The figures are extracted from Oscar Liang Website.

1. Go to the Motors tab in the software.

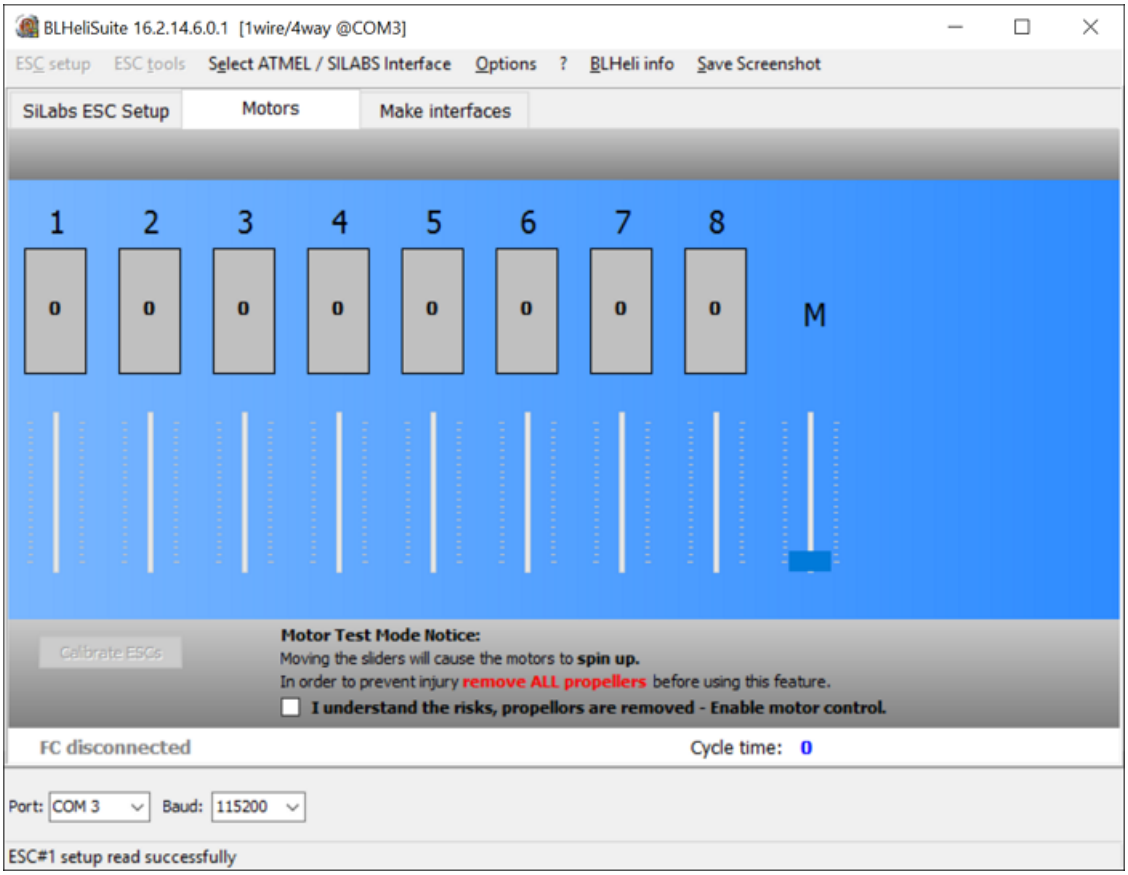


Figure 4: BLHeliSuite32 Calibration

2. Unplug the battery.
3. Check "I understand the risks,..." box.
4. Push the button "Calibrate ESCs".
5. Connect the Battries and Press OK! Then your motors will beep for a few seconds.
6. If it was needed, the drag the slider all the way down after the beeping has completely stopped.
7. The process is completed.
8. For additional check, you can use the slider and see if all the motors spin at the same rate.

## 2.3 Explanation on Parameters of BLHeliSuite32 for ESC Setup

There are various parameters in the setup section (tab) in BLHeli software which can be tuned to achieve better performance of ESC. Some of these features must be tuned for our specific type of motors, while the rest can be left as default.

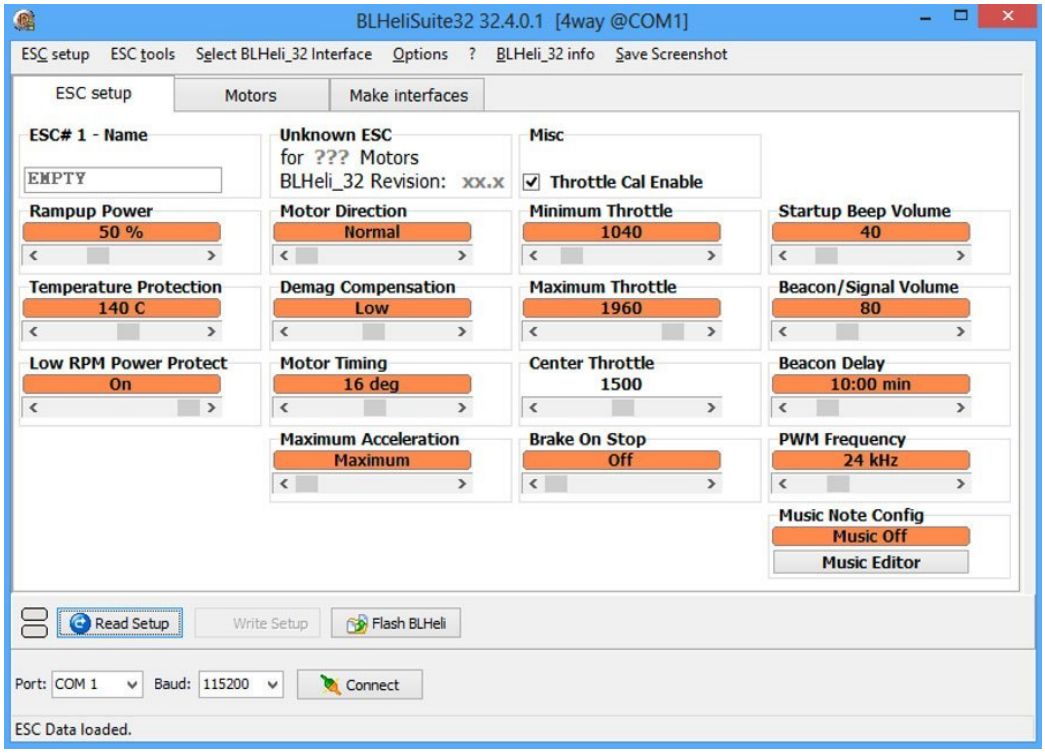


Figure 5: ESC setup tab on BLHeliSuite32 (default configuration, not the one that is being used in the lab)

1. **Motor Timing**:

   In brushless electrical motors, to make the rotor spin, a magnetic field with switching polarity is imposed. Different polarity is obtained by using multiple coils inside the motor and the main purpose of *Motor Timing* is to prepare the magnetic field a little earlier.

   *"Motor timing adjusts how far advanced the ESC drives the magnetization of the field windings when compared to the magnetic fields of the permanent magnets. Almost all mini-quads run medium or high timing for maximum thrust."*[2]

   Basically, by increasing the timing, the power will increase while the efficiency might reduce. It depends on various factors inside the motors and there is not a golden optimal value for everyone. Different motors with different functions will lead to different motor timing requirement. *"The optimal motor timing value can actually change with motor RPM, therefore you are given the option Auto too. It lets the ESC decide what motor timing to use on the fly."*[3] In addition to angular velocity (RPM), there are other factors that influence the optimal value for motor timing. Motor timing relates to the fact that what is the best time to fire (or boost) the upcoming poles in the motor, to attract the rotor's magnet and make it rotate at a desired speed. Therefore, physical properties of the inductors and also magnetic properties of the rotors play a central role in determining the best value.

   For further illustration, it is helpful to step back a little and delve into how ESC is interacting with a brushless motor[4]. Rotation of the shaft inside motor is due to consecutive activation/deactivation of the coils. As the magnetic rotor rotates, it induces some voltage in non-active coils and it is called Back Electro-Motive Force (BEMF) which opposes the main stream of magnetic field. ESC measures this BEMF to know the appropriate time to activate the following coils. *In the ESC it is measured in degrees, though that is a virtual measurement, not a real angle. The assignment seems*

[2]Quoted from Propwashed website; from here
[3]Quoted from Oscar Liang website; from here
[4]Interested readers are encouraged to look at here

*arbitrary, but it is related to the number of the phases in the motor.* In majority of DC brushless motors, this voltage reaches its peak at 0 degree measurement and drops back at 30 degrees.

Finally, advancing the motor timing by 30 degrees means that energizing (activating) the non-active coils occurs right after 0-degree-peak is detected. Reversely, 0 motor timing advance causes the ESC to wait for the BEMF wave to decay completely and then energizes the non-active coils.

Generally, the higher the motor timing is, the less chances of de-sync exists. Using high timing is a conservative way to make sure that you get the best torque and RPM out of your motor. However, if the motors (propellers) get blocked and stop spinning, higher values of timing will increase the risk of breaking down in the motors since it is pushing more current through the coils. If you set the timing on *Auto*, the ESC tries to get find the best timing at each RPM and adjusts itself over time. On the other hand, by setting on Auto, you may not be able to achieve the highest power from the motor.

Having a look at the data sheet for our specific motors in fig.(6), the suggested motor timing is between 5-10 degrees. We decided to choose a value that does not exceed this range and set it to 8. Nevertheless, different values can be examined and used until you get the desired performance[5].

| Cobra CM-2206/30 Motor Specifications | |
|---|---|
| Stator Diameter | 22.0 mm (0.866 in) |
| Stator Thickness | 6.0 mm (0.236 in) |
| Number of Stator Slots | 12 |
| Number of Magnet Poles | 14 |
| Motor Wind | 30 Turn Delta |
| Motor Kv Value | 1400 RPM per Volt |
| No Load Current (Io) | 0.52 Amps @ 12 Volts |
| Motor Resistance (Rm) per Phase | 0.185 Ohms |
| Motor Resistance (Rm) Phase to Phase | 0.123 Ohms |
| Maximum Continuous Current | 17 Amps |
| Max Continuous Power (3-cell Li-Po) | 190 Watts |
| Max Continuous Power (4-cell Li-Po) | 250 Watts |
| Motor Weight | 36.5 grams (1.29 oz.) |
| Outside Diameter | 27.0 mm (1.063 in.) |
| Motor Shaft Diameter | 3.00 mm (0.118 in.) |
| Prop Shaft Diameter | 5.00 mm (0.197 in.) |
| Motor Body Length | 16.5 mm (0.650 in.) |
| Overall Shaft Length | 35.3 mm (1.390 in.) |
| Motor Timing | 5-10 degrees |
| PWM Frequency | 8-20 KHz |

Figure 6: Cobra Motor Specifications: CM-2206/30

2. **PWM Frequency:**

   This parameter relates to the controlling signal from the ESC to the motors which is by default set on 24kHz in BLHeil software. This has nothing to do with the flight controller loop-rate or the signal that is being received by the ESC from the flight controller. Generally, increasing this frequency causes the motor to run smoother with less noise in expense of reduced power or torque.

   For our project, based on the specifications in fig(6), we set this parameter equivalent to 16 kHz.

3. **Minimum/Maximum Throttle:**

   The available range for input throttle signal is between 1000 and 2000. It has been said that a minimum difference of 140 must be kept between thee maximum and minimum limits. It is usually left as the default settings suggest, although it should be consistent with the configuration on the flight controller and probably the code you have written to send the control signals.

4. **Rampup Power:**

[5]The default value, 16 degrees, can be used for instance.

This parameter affect the performance of the motor when it wants to start from zero RPM and limits the variations in power when the throttle is to change abruptly. In applications, similar to ours, which the motors are not subjected to large rotational inertia, the default value works fine. In fact, this parameter influences the ESC's prediction of power required to start rotating and large values of rampup power might lead to some current spikes at initial phases.

5. **Demag Compensation:**

   This feature helps to prevent any problems regarding synchronization in the motors. Usually, the default value works fine, however, in case of sync-ing issues, tuning this parameter can be helpful.

   *"In all versions of BLHeli, Demag Compensation makes some small changes to the commutation methodology that helps recover from potential desync situations. The demag setting checks for the BEMF signal more rigorously for bad reading and will suspend energizing the coil of a commutation to make the next 0 crossing easier to detect. Demag Compensation will also attempt to drive the motor without readings similar to the method used when the motor is first starting up, using the last known RPMs to predict the phase positions. The high and low settings change the threshold that BLHeli uses or the number of bad BEMF reads that it is willing to accept before compensating, and turning it off means BLHeli will not attempt correct for bad readings at all. This functionality is the same in all revisions of BLHeli."*[6]

## 2.4 Writing Custom Configuration on ESC via. BLHeliSuite32

Previously, various features of BLHeli software was introduced which can be modified bu the user. In this section, the steps to write the custom configuration on ESC board is explained.

1. Connect the ESC and read setup (same as section (2.1)).
2. Make sure that all the motors has been recognized by the software (See section (**??**)).
3. Choose the parameters in ESC setup. Read section (2.3).
4. Click on Write Setup and write the configuration on the board.

## 2.5 Flight Controller (CleanFlight) Firmware Update/Flashing

### 2.5.1 Driver Installation

Before using cleanflight software, please make sure that the required drivers are installed on your PC. There are three packages as following which are suggested to be installed.

1. **CP210x Drivers**: Get it from here.
2. **STM USB VCP Drivers**: You can download it from here.
3. **Zadig** for Windows USB driver: Get it from here.

### 2.5.2 Normal Flashing

1. Connect the flight controller to PC.
2. Open Cleanflight Configurator software.
3. Click on disconnect if connected automatically.

[6] Excerpt from Mini Quad Test Bench website, here.

4. Click on Firmware Flasher tab.

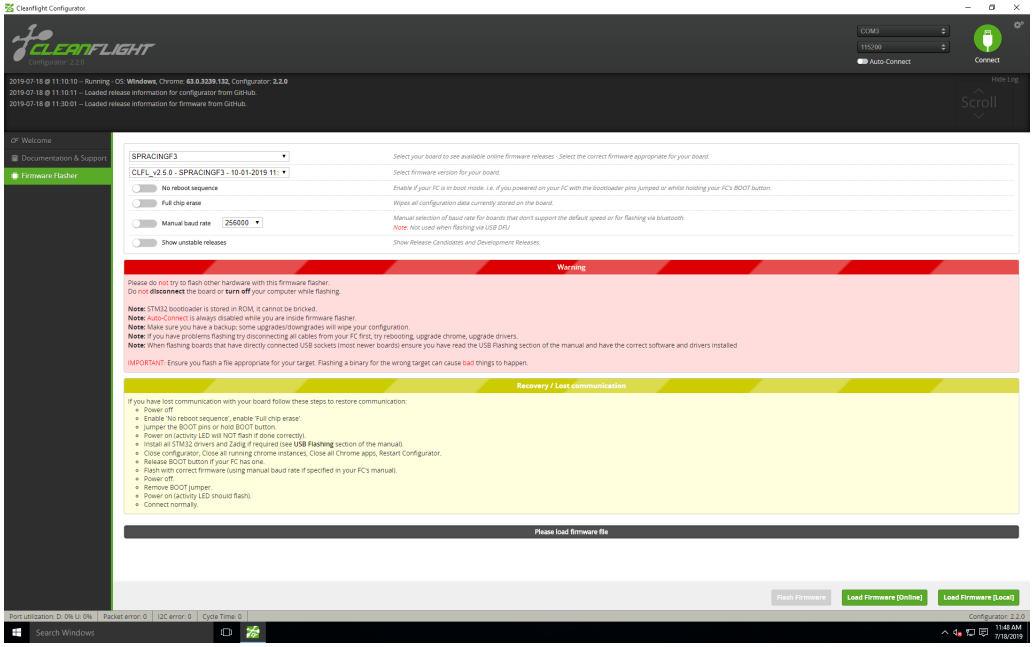


Figure 7: Cleanflight Firmware Flasher

5. Choose the appropriate board model for firmware. In our case: SPRACINGF3
6. Select a suitable version for the firmware.
   (a) If you have not calibrated or flashed the ESC and you wish to use BLHeliSuite 32, you need to select the latest version for the firmware.
   (b) If your ESC configuration and calibration are ready and you wish to run the code, make sure that you choose a compatible version of firmware. In our case, we needed to select CLFL_v2.1.0 - SPRACINGF3 - 30-07-2017 or any other 2017 versions should work fine.
7. Make sure that the Internet is connected and click on Load Firmware [Online].
8. In case of first-time flashing, enable the Full Chip Erase checkbox.
9. Make sure that the correct COM port is selected.
10. Click on the Flash Firmware.
11. When the progress bar becomes green and reads Programming: SUCCESSFUL you are done!

### 2.5.3 Recovery/Reset Mode

1. Disconnect the lfight controller usb.
2. Connect two BOOT jumper pins on the board.
3. Close all the windows and restart the configurator.
4. Enable No reboot sequence on cleanflight interface.
5. Flash to the desired firmware.
6. Disconnect the USB.
7. Power it on again and check if it connect normally.

## 2.6 Cleanflight Configuration

Cleanflight interface provides a sophisticated and relatively elaborate settings which can be tuned based on user's expectations. Not all of the sections need to be changed for our purposes and in this section we are only related part will be investigated.

### 2.6.1 Setup

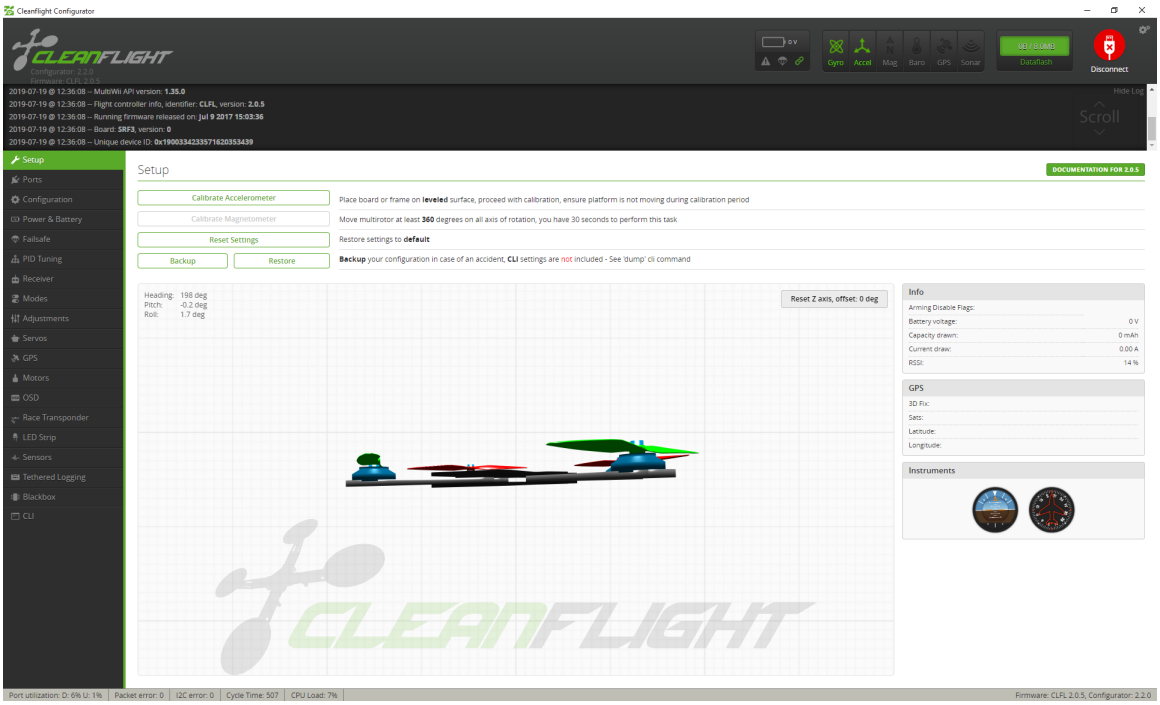


Figure 8: Cleanflight - Setup Settings

1. Make sure that the drone (and hence, the flight controller) is placed on a horizontal plane.
2. The values of Pitch and Roll shown at the corner must be zero (almost zero). Otherwise, the accelerometer needs to be calibrated.
3. Click on Calibrate Accelerometer. Wait until the procedure is completed. Afterwards, the values of roll and pitch must be close to zero.

**Note:** The gyro sensor inside the flight controller is sensitive to mechanical impacts and it is highly recommended to re-calibrate the gyro after crashes.

### 2.6.2 Ports

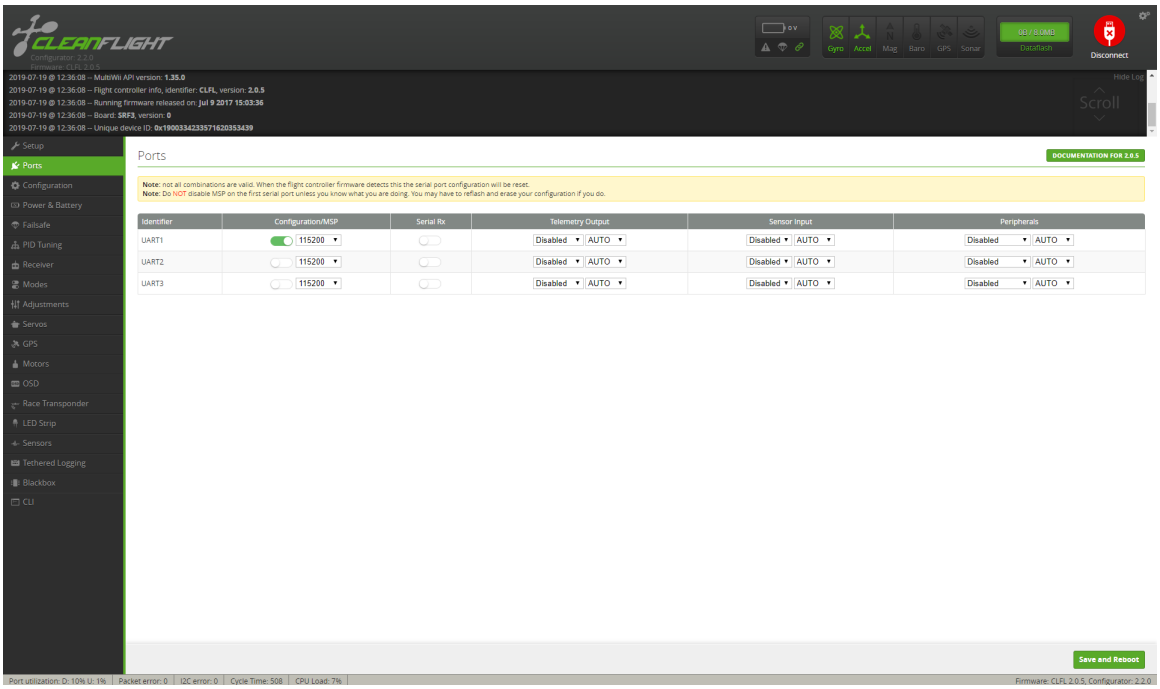


Figure 9: Cleanflight-Ports Settings

| Value | Identifier | RX | TX | 5v Tolerant | Notes |
|---|---|---|---|---|---|
| 1 | USART1 | PA10 | PA9 | YES | Internally connected to USB port via CP2102 IC. Also available on a USART1 JST connector and on through hole pins. |
| 2 | USART2 | PA15 | PA14 | YES | Available on USART2 JST port only. |
| 3 | USART3 | PB11 / IO2_3 | PB10 / IO2_4 | NO | Available on IO_2, USART3 JST port and through hole pins. |

Figure 10: SP Racing F3 Serial Ports

The configuration that has been used for our application in the lab is shown in the figure (9). The table which is being shown in figure (10) is extracted from cleanflight website and states that UART1 is internally connected to USB port. Therefore, by enabling UART1, we will be able to use either USB connection or wireless connections without changing the settings.

### 2.6.3 Configuration

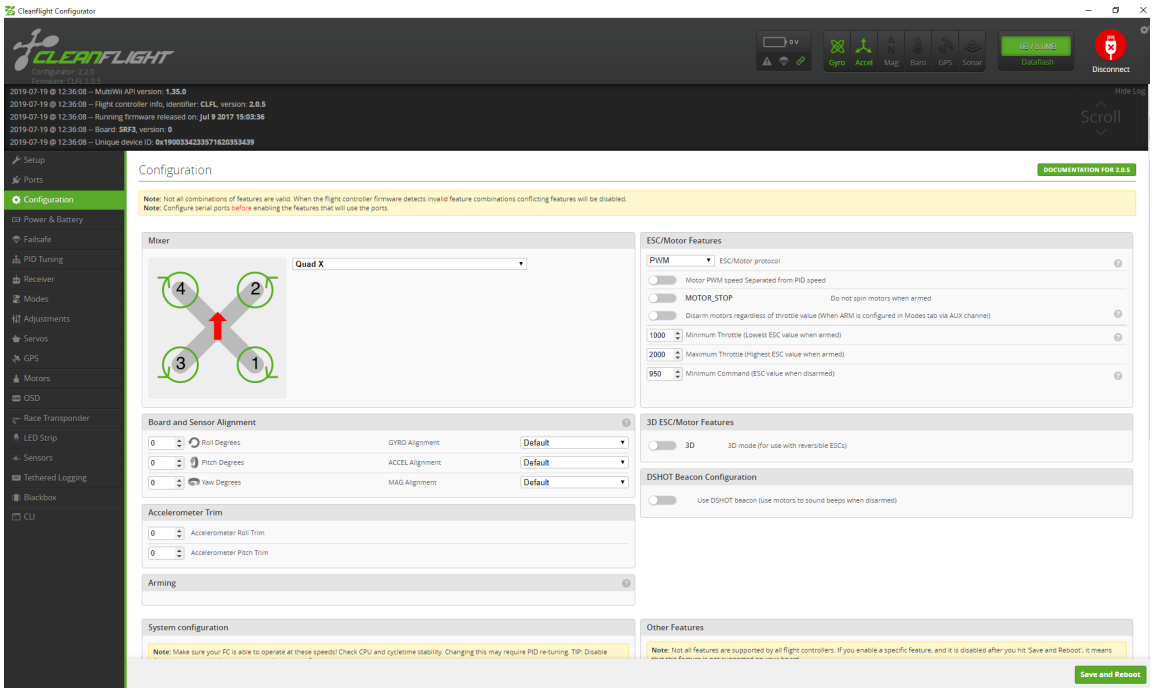


Figure 11: Cleanflight - Configuration Settings - Part 1

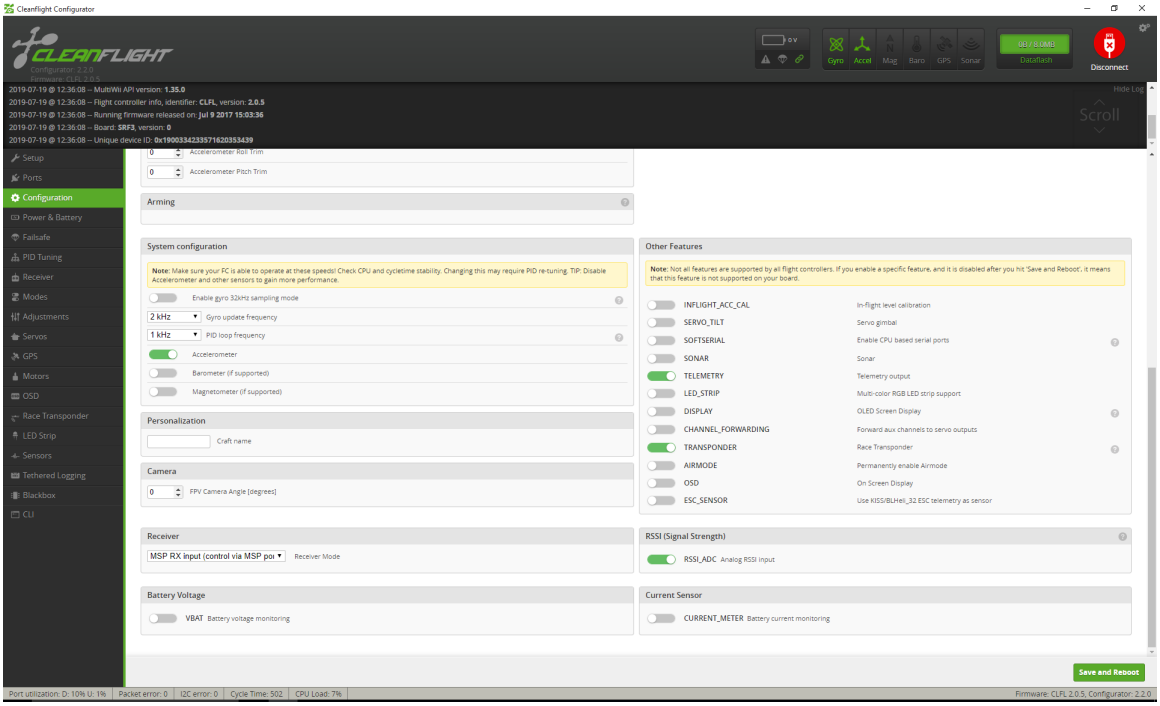


Figure 12: Cleanflight - Configuration Settings - Part 2

### Mixer

For a quadcopter, the confiduration of the propellers and their direction should be chosen a shown in the fig. (11). It must be noted that the red arrow on the picture must be aligned with x-axis denoted on the clean flight. In our specific assembly, the directions of the props and the flight controller are as following:

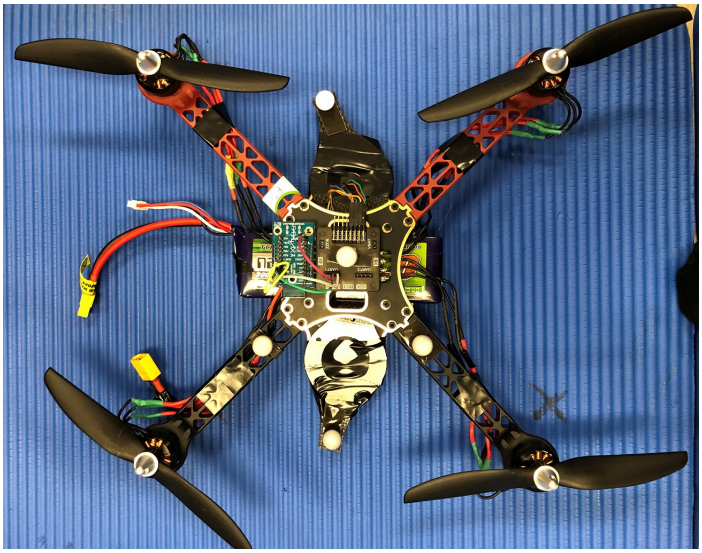

Figure 13: The arrangement of propellers and the flight controller on a quadcopter

For more illustration, in the next figure the direction of x-axis on the flight controller has been specified (inside the red circle):

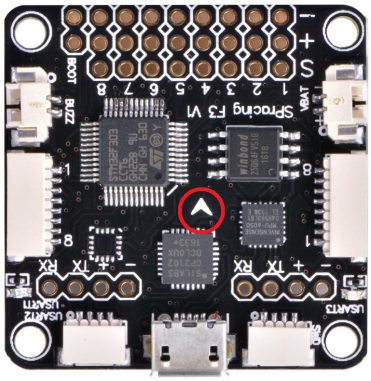


Figure 14: Direction of the accelerometer of the flight controller in red circle

### Board and Sensor Alignment

These options can be set in the case if you have some misalignments in your structures relative to default horizontal plane. In our setup, we left all the alignment values zero as default.

### ESC/Motor Features

From **ESC/Motor Protocol** menu, you are able to chose the appropriate protocol for your setup. You can find more details about different ESC protocols in appendix, however, in our case, Pulse Width

Modulation (PWM) was selected.

### Motor PWM Speed Separated from PID Speed

By Enabling this option, you can tell the flighgt controller to update the motor throttle values at a rate different from the loop-rate of the PID controller. This results in a de-synchronization between flight controller PID and the PWM signal that is being sent to the motors. To generate the PWM for motors at a separate rate, the flight controller uses an embedded hardware PWM generator which influences the performance of the CPU inside the controller. Basically, by delegating the task of PWM signal generation, the flight controller's CPU's work load will decrease by a few percent.

Some of the developers have claimed that setting the Motor PWM frequency at a higher value than PID loop-rate, for instance twice the PID rate, can help the accuracy of the PWM signals. Their argument comes from the fact that PWM is analog and it can be corrupted by interference and noise, hence, sending the throttle value multiple times at each single loop of controller decreases the chances of interruptions or data loss. On the other hand, using a rate lower than PID loop seems not to be beneficial and is not suggested. In our case, we did not use this option and preferred to stay with synced PID and Motor PWM.

### MOTOR_STOP

As the explanation suggests, enabling this option prevents the motors from spinning after getting armed. If you enable this option, you can see that another option appears as depicted in fig. (15).

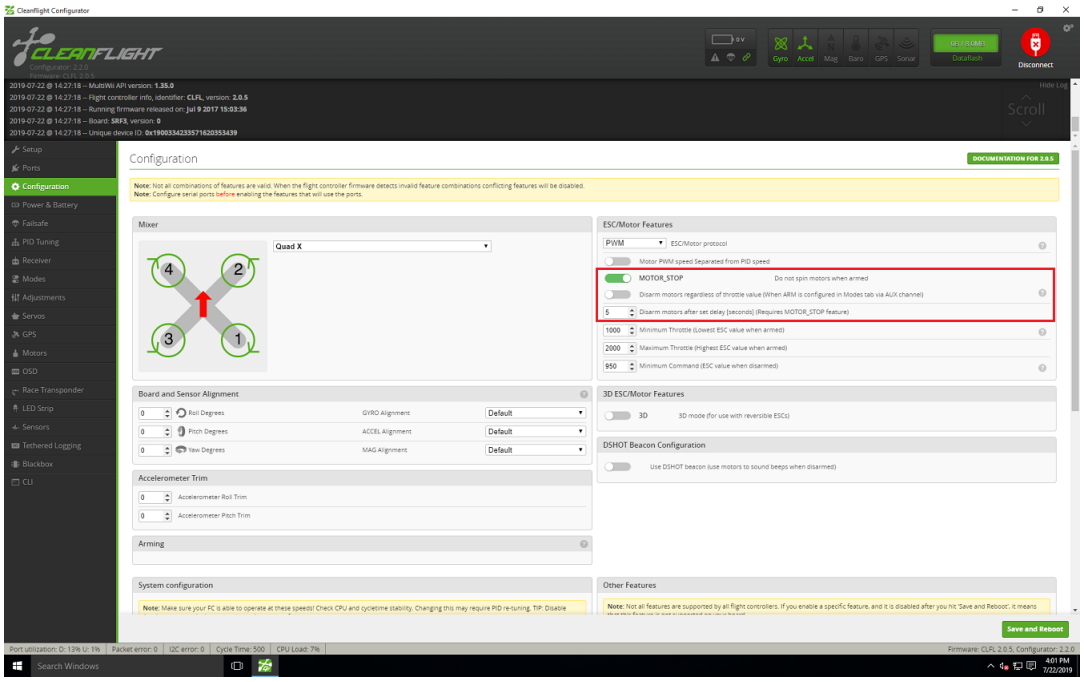


Figure 15: Motor Stop Option in Cleanflight

You can set a time delay in which if the controller did not receive any commands from the user, disarm the controller. For example, the setting inside the red box in figure (15) suggests that after getting armed, the controller waits for five seconds and it did not receive anything, it just disarm the controller.

There is another switch which states: "*Disarm motors regardless of throttle value.*" Disabling this switch means that the flight controller will not get disarmed unless all the throttle commands become zero. It is suggested to leave it disabled to avoid accidental crashes and stops in the middle of flights.

### Minimum/Maximum Throttle - Minimum Command

In this section, you can set the values which determine the maximum and minimum value of the signal sent for throttle. A normal choice would be 1000 for minimum and 2000 for the maximum throttle. Moreover, minimum command value specifies the value sent by the ESC to the motors when disarmed or armed with MOTOR_STOP on. It should stop your motors from spinning and therefore, **Minimum Command must be less than Minimum Throttle in BLHeli**. We had chosen this value to be 950.

### 3D ESC/Motor Features

This mode enables the quadcopter to fly in an inverted orientation and therefore, it requires special configurations on mechanical assembly and software settings. For instance, bi-directional motors and properellers are needed. Moreover, the ESC must be reversible and additionally, you need to activate bi-directional mode on ESC firmware (BLHeli). Anyway, in our setup we did not need this mode and left it disabled.

### System Configuration

There are two principal parameters that need to be known and determined appropriately in this section.

1. Gyro Update Frequency: This is the rate at which the controller samples measurements from the gyro sensor embedded inside it.
2. PID Loop Frequency: This rate specifies how often a controller loop runs.

The PID controller uses the measurements from gyro sensor to generate suitable control inputs based on the error with desired values of acceleration. Consequently, the controller's loop must not be faster than gyro update frequency.

In addition to gyro update frequency, there are multiple factors that can possibly affect your choice of PID loop-rate. One of the main factors is ESC protocols which has been used in for communication between the flight controller and ESC. Generally, it makes more sense to establish a protocol faster than the controller. In other words, if the frequency of transmitting data between ESC and the flight controller is higher than the rate at which the control inputs are generated, it reduces the chances of miscommunication or data corruption. The ESC will be receiving multiple throttle values per each single control loop and this will increases the robustness of the system respect to possible errors in data transferring.

*In practice, you really shouldnt run anything above 3.8KHz looptime with Oneshot125. This ensures a small gap in the PWM signal to allow the ESC identify the signal correctly, otherwise invalid signals can cause an ESC to shut down and malfunction. Oneshot125 is too slow for 8KHz, because it has a latency of 125uS to 250uS (which is 8KHz to 4KHz), it can only handle 4KHz looptime in theory. Any protocol that is faster than Oneshot125 is capable of managing 8KHz looptime, such as Oneshot42 and DShot300, while for 32KHz looptime you will want to use Multishot, DShot600 or other faster protocols.*[7]

In our setup, we used 2kHz for gyro and 1kHz for PID frequency. There are three other options that you can determine which measurements you will be reading such as gyro, barometer, and magnetometer where only the first option is sufficient and also necessary for our application.

### Receiver

This part relates to the way the communication is done to send commands to the drone. There are several types of receivers which can be utilized via Cleanflight firmware. If you click and drop the menu down, you will see there are the following features:

[7] Based on Oscar Liang Explanations, here.

1. RX_PPM
2. RX_SERIAL
3. RX_PARALLEL_PWM
4. RX_MSP

Only one receiver feature can be enabled at a time. In our setup, MSP protocol has been used.

### 2.6.4 Failsafe

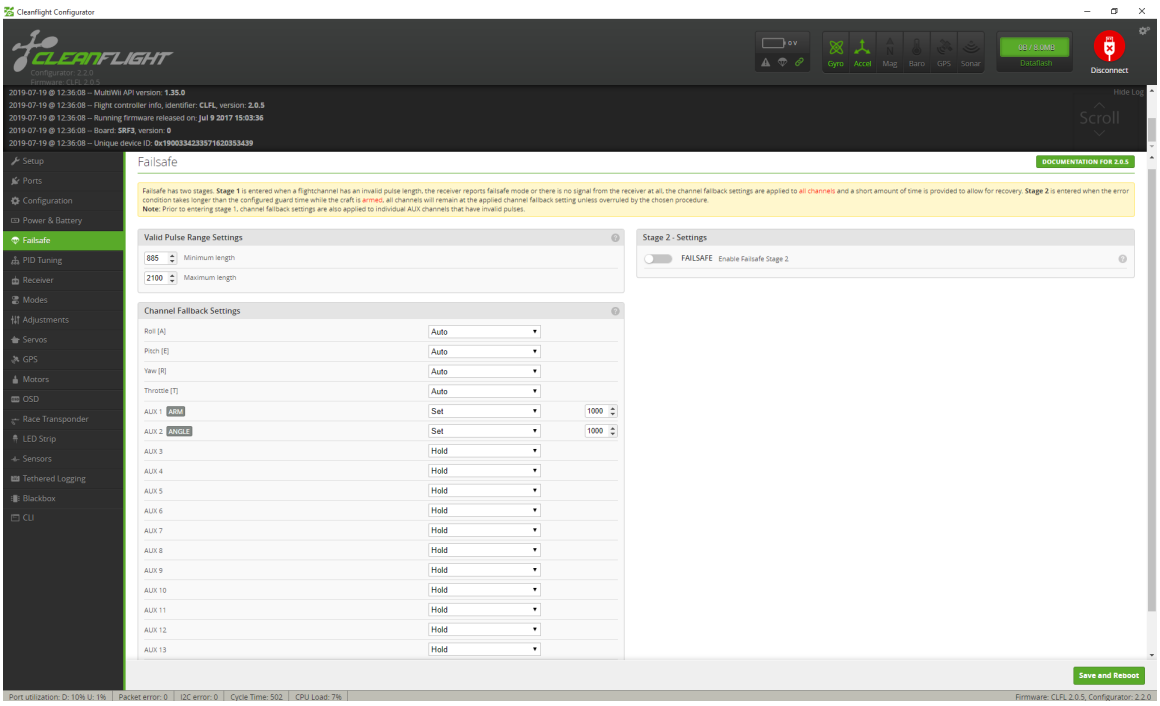


Figure 16: Cleanflight - Failsafe

#### Valid Pulse Range Settings

As the section name suggests, you can decide to choose an interval for a valid received pulse range. For example, in figure (16), the minimum length is set to 885 and the maximum is 2100. Therefore, if the flight controller receives a signal out of this range, it knows that the signal is not trustworthy and **it will initiate the failsafe mode**.

#### Channel Fallback Settings

In this section, you can choose the first stage of cleanflight failsafe and it will be initiated if the signal was not valid. For each channel, there is drop-down menu that gives you three different options to choose. **Hold** option means that for that specific channel, the last valid value would be held. Therefore, setting the channels on Hold may be dangerous and it is not generally suggested. **Auto** means that the values of Roll, Pitch and Yaw will be maintained in the middle of the range and the throttle will be kept as low as possible. Additionally, you can set a fixed value by choosing **Set** option and determining your desired value in case of invalid signal.

### Stage 2 - Settings

This stage is not enabled by default and by switching it to enabled, a setting similar to what has been show in the next figure will be displayed.

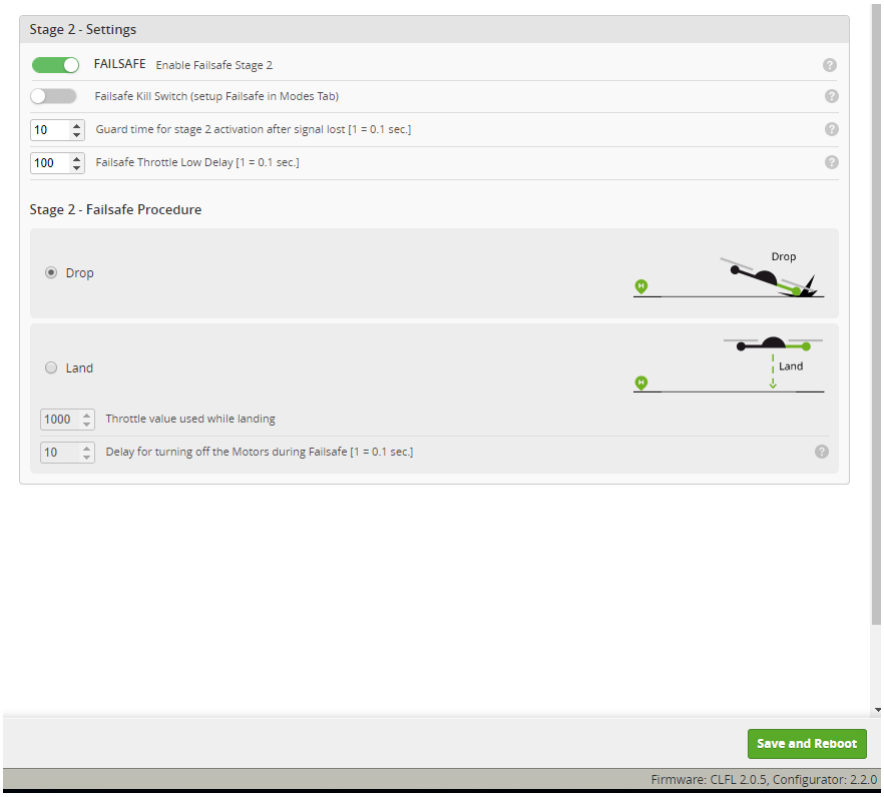


Figure 17: Cleanflight failsafe stage 2 enabled

**Failsafe Kill Switch** option provides you with the chance to set a switch to act as a *kill switch* and bypassing the selected failsafe procedure. If you enable this option, you will need to set a switch for this purpose in Modes Tab. **Guard time** indicated the time when the controller waits after receiving an invalid signal to enable the second stage of failsafe. For instance, the number 10 correspond to 1 second for activation of the second stage of failsafe. The **Failsafe Throttle Low Delay** is used to determine the time for the controller to disarm after losing the signal. Finally, there are two scenarios for failsafe procedure in this stage which are called **Drop** and **Land** procedure.

Our setup is the same as figure (16) and we did not enable the second stage of failsafe. There is a fact based on our application and environment, enabling the second stage is not necessary and also it increases the chances of arming problems with the flight controller, see section (**??**).

### 2.6.5 PID Tuning

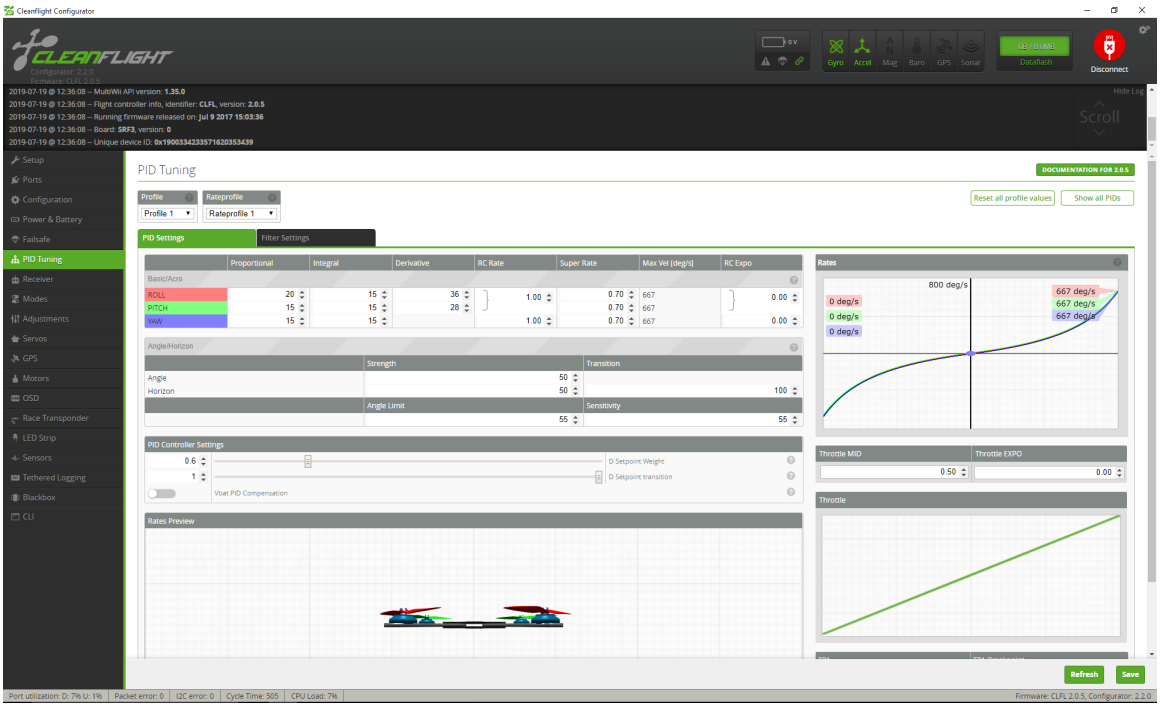


Figure 18: Cleanflight PID Tuning

Generally, PID gains are needed to be tuned for each specific device separately. These gains depend on different parameters and also their optimal value is pertinent to what is your desired (optimal) behavior; i.e. from your perspective. On the other hand, there are some useful tips that con be taken in order to achieve a good design.

**Hints**:

1. The gains can be different around each axis since the inertia moment may have different values around each axis. For instance, our configuration in figure (13), shows that the value of inertia around x-axis (Roll) is probably greater than other axis (The battery is perpendicular to x-axis).
2. Higher P-gain results in sharper movements. It becomes faster but the amount of overshoots will increase and higher-frequency oscillations are expected.
3. Low values of P-gain might show fewer oscillations, however, the general performance would be sloppy.
4. I-gain determine the behavior of the system against disturbances and offsets such winds.
5. *If you notice some drifting without user command, then increase it. When I-gain is too low you might find yourself having to correct the quads flying path a lot more with your sticks, especially when you are active with the throttle.*[8]
6. High values of I-gain will cause the system behave stiffer, unresponsive and slower.
7. Large D-gains can intensify the noise in the system and cause it to vibrate uncontrollably.
8. However, D-gain acts as a damper and it is used to alleviate the effect of proportional gain in overshoot problems and oscillations.

[8]Borrowed from Oscar Liang explanations, here.

### 2.6.6 Receiver

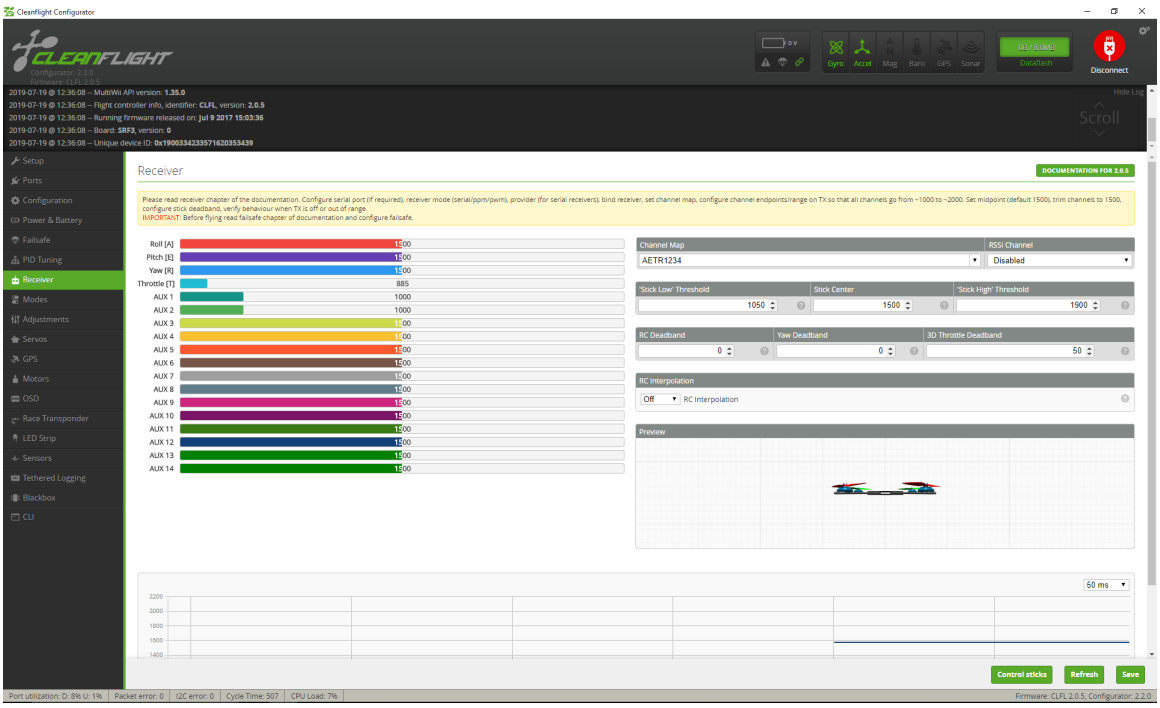


Figure 19: Cleanflight - Receiver Settings

1. **Channel Map:** AETR1234
   This section is based on the direction of the motors' rotation which was set also in figure (11).

2. **RSSI Channel:** Disabled
   RSSI is a measurement of signal strength and is very handy so you know when your aircraft is going out of range or if it is suffering RF interference.
   Some receivers have RSSI outputs. 3 types are supported.

   (a) RSSI via PPM channel

   (b) RSSI via Parallel PWM channel

   (c) RSSI via ADC with PPM RC that has an RSSI output - aka RSSI ADC

   In our case, since we are working in a relatively small space and the interference is not considered as an effective factor on quad's performance. Therefore, it can be set to disabled. For more information, see here.

3. **RC Interpolation:** Off
   Interpolation of Rc data during looptimes when there are no new updates. This gives smoother RC input to PID controller and cleaner PIDsum.

Other values such as stick center and thresholds may set as depicted in figure(19). Moreover, by clicking **Control Sticks**, a panel will show up on your screen and it is basically the same as control sticks on a usual RC. If you connect the battery and **Enable** the sticks, you can easily test the performance of the sticks and the corresponding AUX channels.

### 2.6.7 Modes

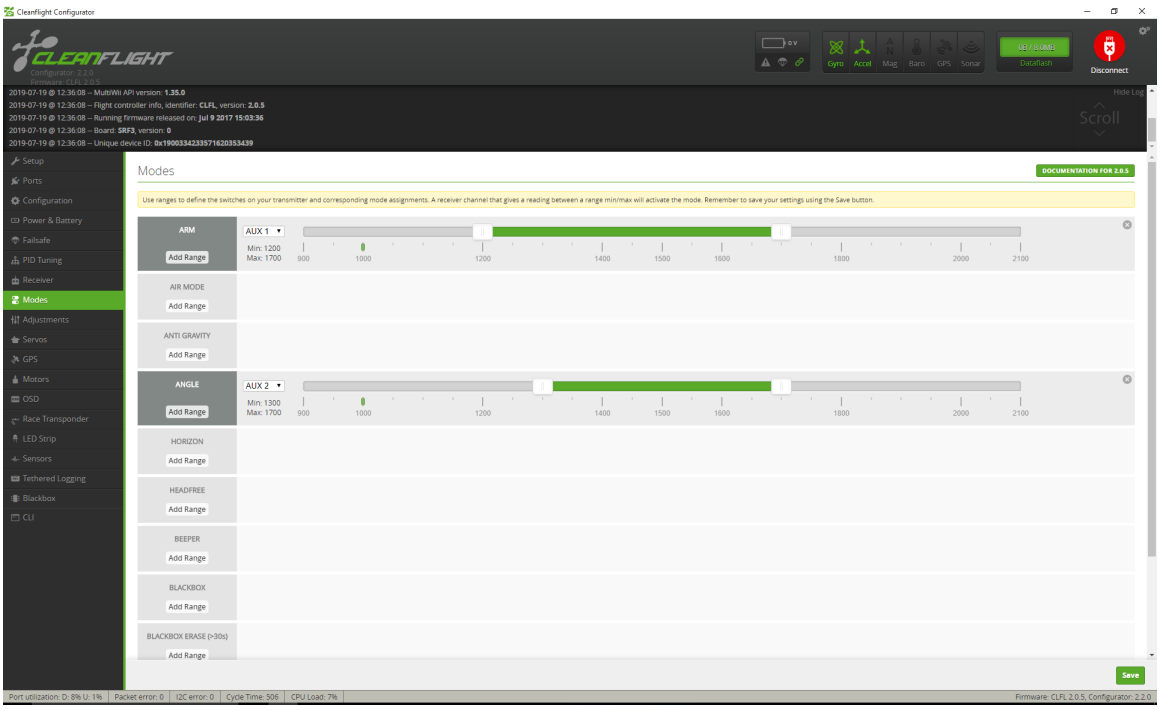


Figure 20: Cleanflight - Modes Settings

*There are various modes that can be toggled on or off. Modes can be enabled/disabled by stick positions, auxillary receiver channels and other events such as failsafe detection.* You can check the full description from this link.

In our setup, only **ARM** and **ANGLE** need to be set. Based on the written code, the value of 1500 will be sent to activate these two modes. Hence, the range must include this specific value and it is suggested to set it as in figure (20). In addition, pay attention to the AUX channel which is assigned for each mode.