Stephen Dade

MHV Quadcopter Workshop Manual 2

July/August 2013

Table of Contents

Objective

To design and build a quadcopter for a workshop to be held at the Make Hack Void Hackerspace (Canberra). It is designed to be:

- (Relatively) Cheap
- Commonly available spare parts
- Open source or hackable hardware/software where available
- Easy for beginners to build
- Strong/tough enough to withstand beginner pilots
- Able to carry a 200g payload (such as a small camera)

What to bring to the Workshop

- Laptop (and charger!). Confirm that the APM Mission Planner software (see the Ground station software section) runs on your laptop beforehand.
- MHV will provide everything else.

Note the workshop will require basic soldering. If you haven't soldered before, we can teach you during the workshop

The hardware

The following list includes all parts (and some tools) required to build the quadcopter. Note you may need a soldering iron for electrical work. Other hardware tools should be in your shed (or local hackerspace!)

The list can be found at https://canberrauav.readthedocs.org/en/latest/quadcopter-workshops/quadcopter-workshop-2-0/parts-list.html for the list of tools.

Total cost of all parts is $730.

The software

Ground Station

The standard GCS is the APM Mission Planner

http://ardupilot.com/downloads/

This will run on Windows XP/Vista/7 or Ubuntu (or similar). Linux users need to install the mono framework.

The program does require a moderately powerful laptop/PC to run on. It will barely run on most netbooks. An active internet connection will be required to use some features.

The APM Mission Planner is frequently updated. Fortunately, it includes auto-update software.

More advanced users can use MAVProxy (a command line GCS) instead:

http://tridge.github.io/MAVProxy/

MAVProxy is compatible on Linux and Windows platforms, and uses the Python framework.

An alternative option is APM Planner 2. It is based off the qGroundControl software, with APM-specific improvements. At the time of writing it is still in beta. APM Planner 2 can run on any Windows, Mac or Linux systems using the Qt framework. It is a GUI based ground station and has a fairly small impact on system resources. It is available at:

http://firmware.diydrones.com/Tools/APMPlanner/

The telemetry radios can be (via the supplied cable) connected to a mobile phone or tablet running Android 4.1 or later. Look for the *Droidplanner* app on the Google Play store.

Autopilot

We use the ArduPilotMega (APM):

https://github.com/diydrones/ardupilot

The APM mission planner includes the APM software in it. This software is being frequently updated with bug fixes, better navigation code and more features. It is recommended to update when a new version is released.

Note that the Pixhawk board is used for this workshop. When uploading firmware ensure the Pixhawk version of the APM software is selected.

A note about terminology: Pixhawk refers to the *physical board based* on the ARM chip. APM refers to the *software* running on the board. To make things more confusing, APM also refers to the physical boards based on the ATMEGA2560 chip.

Due to the 2 hardware boards, there are two builds of the APM software:

- ATMEGA board
- Pixhawk (or PX4) board

It should be emphasised that the ATMEGA and Pixhawk builds run the same APM software, minus the different hardware drivers. The high level navigation, positioning and other features are exactly the same.

For the purposes of this workshop, I will refer to the Pixhawk when referring to the board and APM when referring to the software.

ESC's

The SimonK firmware is compatible with the ESC's used in this project.

https://github.com/sim-/tgy

This firmware increases the update rate of the ESC's, making the ESC (and hence motors) more responsive to commanded throttle changes.

Transmitter

The er9x firmware is built for the Turnigy 9X transmitter.

http://code.google.com/p/er9x/

The er9x firmware makes the GUI far easier to navigate.

Radios

Like the APM, the radios use open-source firmware.

http://code.google.com/p/ardupilot-mega/wiki/3DRadio

The firmware is not frequently updated, so we probably won't need to update this.

Build Instructions

Timing

The build will take several evenings. Flight testing and lessons will take an afternoon.

The timing is:

- **Evening 1** –Introduction. Flashing ESC's, Transmitters and frame assembly. Soldering of power connectors. Simulator setup
- **Evening 2** – Flight electronics installation.
- **Evening 3** – Calibration and testing. Safety Briefing
- **Evening 4** – Indoor flight at Dickson College. Simple flights
- **Afternoon 1/2** – Outdoors flight at a local oval. Training of advanced flight modes

Evening 1

Introduction to Quadcopters

A quadcopter is a flying vehicle possessing 4 identical rotors, evenly spaced around the central fuselage (hub).

First, quadrotors do not require mechanical linkages to vary the rotor blade pitch angle as they spin. This simplifies the design and maintenance of the vehicle. Second, the use of four rotors allows each individual rotor to have a smaller diameter than the equivalent helicopter rotor, allowing them to possess less kinetic energy during flight. This reduces the damage caused should the rotors hit anything. For small-scale UAVs, this makes the vehicles safer for close interaction. Some small-scale *quadrotors have frames that enclose the rotors, permitting flights through more challenging environments, with lower risk of damaging the vehicle or its surroundings.* (Wikipedia)

Components

Electronic Speed Controller (ESC): Regulates power to the motor according to the input throttle level. It also provides +5V power for the flight electronics

RC Receiver: A (usually 2.4 GHz) RC radio receiver on the quadcopter that receives commands from the RC transmitter on the ground. One way link

Telemetry Link: A (usually 915 MHz or 433 MHz) bidirectional link between the flight controller and ground station. Provides current status to the ground station and accepts flight commands to the quadcopter.

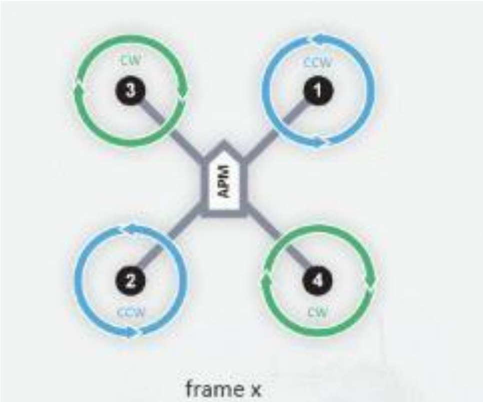
RC Transmitter: A (usually 2.4 GHz) RC radio transmitter used by the pilot to direct the quadcopter's direction and position.

Hub: The central fuselage of the quadcopter. Contains the flight electronics and battery

Arm: The beam that each of the motors sit on.

Generating Thrust

The motors and propellers alternate in direction and pitch, in order to cancel out and horizontal torque. All propeller/motor combinations still produce downwards thrust:



By altering the thrust to each of the motors, the quadcopter can move and turn:

Pitch (forward/back): Decrease thrust to motors 1 and 3. Increase thrust to motors 2 and 4 to maintain altitude. This will pitch the quadcopter forwards, giving it a forward velocity.

Yaw (horizontal rotation): Decrease thrust to motors 3 and 4. Increase thrust to motors 2 and 1 to maintain altitude. This will rotate the quadcopter counter-clockwise.

Roll (left/right): Decrease thrust to motors 2 and 3. Increase thrust to motors 1 and 4 to maintain altitude. This will roll the quadcopter left, giving it a sidewards velocity.

Basic Stats

The basic stats are:

- Top speed: 2.5 m/s vertical, 6m/s horizontal (no payload)
- Max Payload: 400g
- Flight time: 12 min (no payload)

Note quadcopters (particular those < 1kg) are very sensitive to payloads. The payload should be as close as practical to the quadcopter's centre of mass.

Autopilot Modes

The ArdupilotMega (2.9.1 at time of writing) has the following flight modes:

- **STABILISE** – The primary mode. Use the RC sticks to navigate the quadcopter. Otherwise, it will stay level
- **ALT_HOLD** – It will maintain the current altitude. Otherwise, the RC sticks can be used to navigate the quadcopter
- **LOITER** – It will maintain position, heading and altitude at the current point. Altitude can be changed by changing the throttle value
- **RTL** – It will return to its takeoff (where the quadcopter was armed) point. Depending on the mode settings, it will either hover at a 5-10m altitude and wait for the pilot to land it or gradually descend until it lands.
- **AUTO** – It will run through the waypoints currently loaded. The waypoints can be a simple "go to point", "loiter for X min", "change velocity", "land". Note there is not automated take off.

- **ACRO** – Advanced Mode. The RC sticks act as rate controllers rather than position controllers. There is no automatic stabilisation
- **LAND** – lands the quadcopter on the ground at its current point

More modes may become available as new versions of the APM software are released.

Initial charging of the battery

The2 batteries come shipped with a nominal 60% charge. Use the battery charger to charge both the flight battery and RC transmitter battery. The settings on the battery charger should be: 3S LiPo battery type, charge rate 5A for the flight battery and 2.5A for the RC Transmitter battery.

Both the power cable and balance connector should be hooked up to the charger.

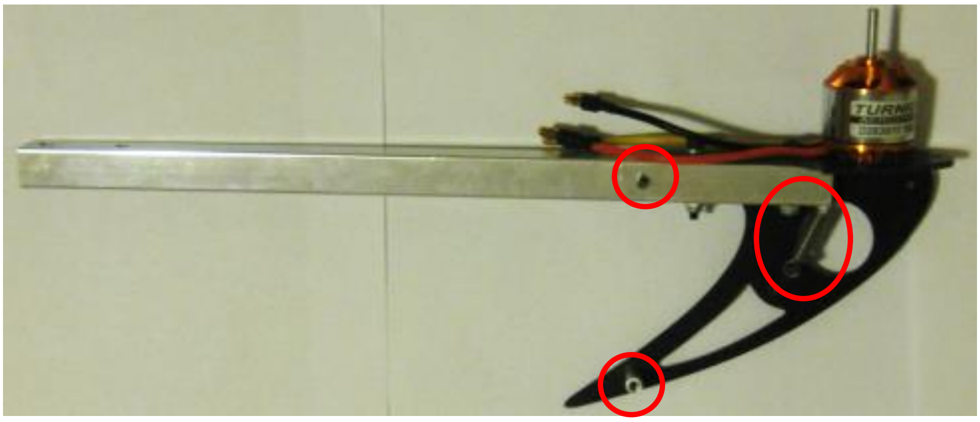
The batteries will be fully charged around the 12.6 V level. The charger will emit a series of loud beeps when it is done.

Frame assembly

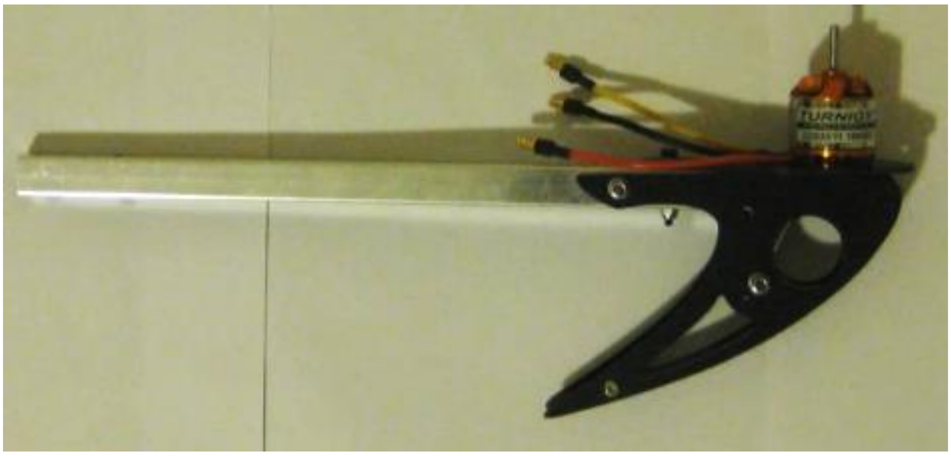
1. Attach the motor mounts to the end of each arm:

2. Use the M3 screws to attach the motors:

3. Attach 1 of the leg struts, along with the spring and screws:

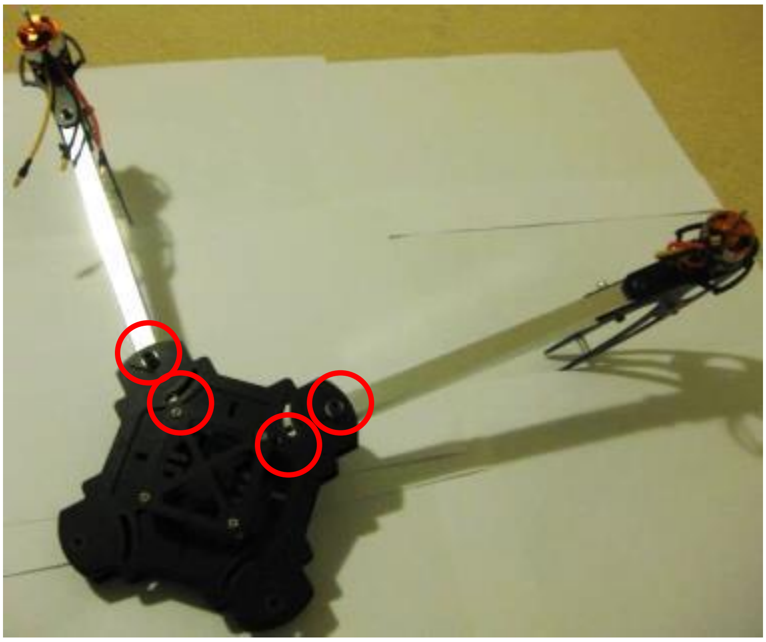


4. Attach the other half of the strut:

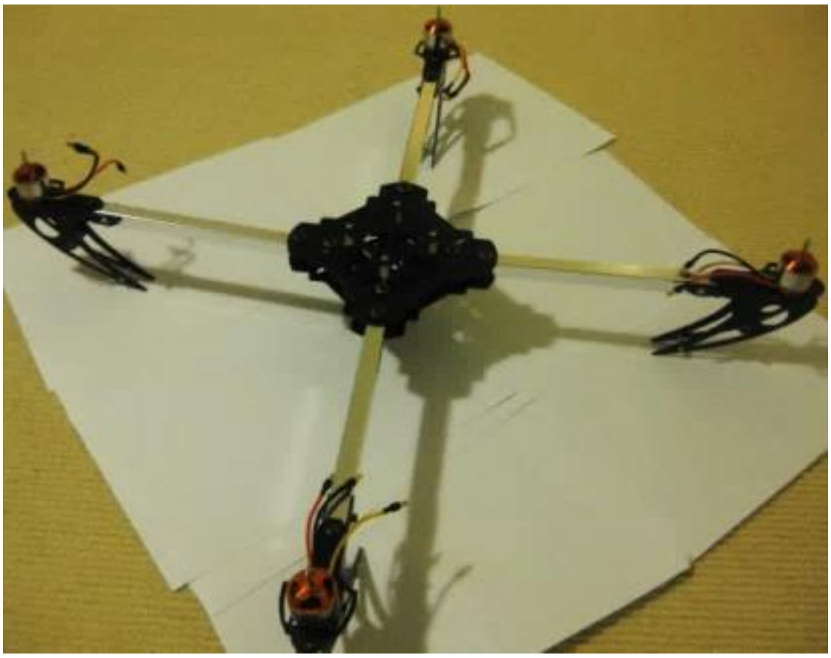


5. Assemble the top of the hub plate. Note there are 3 plates here:

6. Along with the bottom plate, start attaching the legs:



7. Add in the rest of the legs:



Thread the small piece of Velcro in a loop through and underneath the bottom hub plate. It should form a harness for the battery.

Evening 2

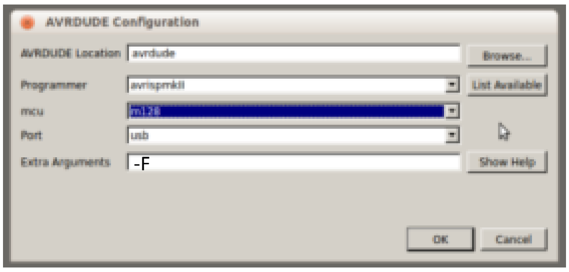
Transmitter Firmware Upgrade

The firmware and settings file for the Turnigy 9XR can be found at https://canberrauav.readthedocs.org/en/latest/quadcopter-workshops/quadcopter-workshop-2.html. Download these files and use then as needed in the following section. The firmware can be found at http://code.google.com/p/er9x/source/browse/trunk/er-128.hex

Use the AVR programmer to connect the Transmitter's 6-pin connector to the laptop's USB port:

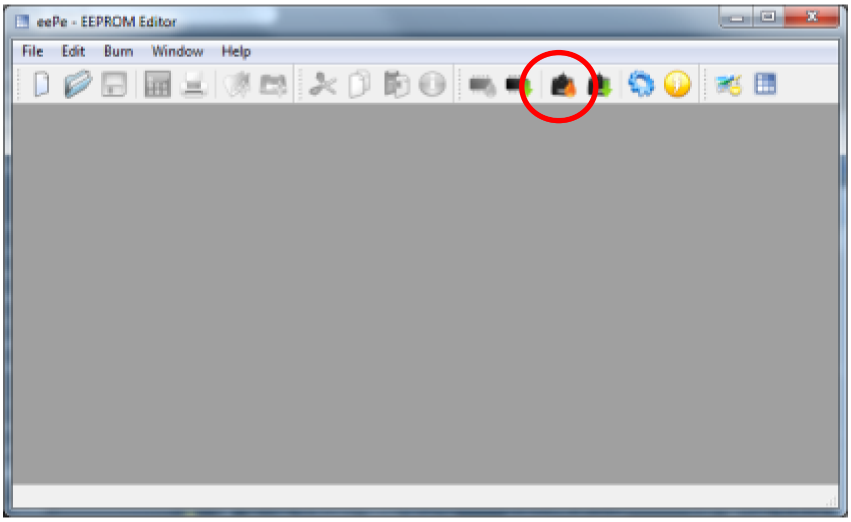
1 - The RC Transmitter connected to the AVR Programmer

Connect the USB side to a laptop and open up Epee (a `sudo` may be required under Linux). Go to Burn -> Configure. Change the programmer to `avrispmkII`, mcu to `m128` and add a "`-F`" to the additional options.



2 - eePe programmer setup

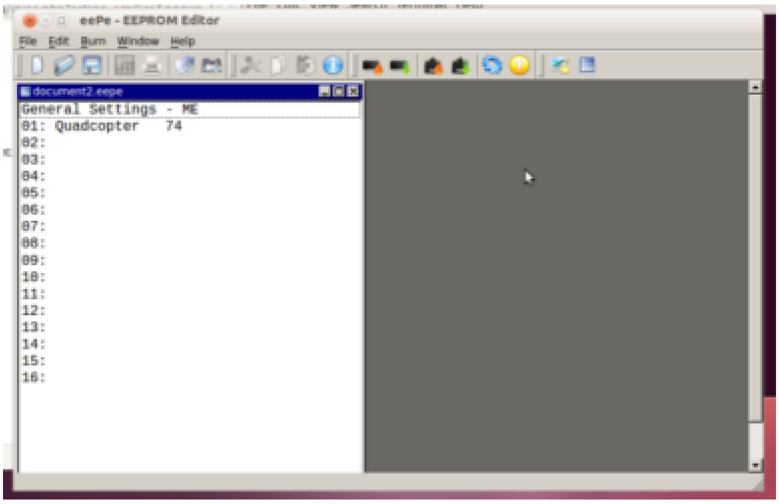
Press the "burn" button on the main screen and select the supplied firmware file to upload.



3 - The eePe software on Windows

Wait for the program to complete and confirm that there were no errors during the flashing process.

To upload the settings file, go to `File -> Open` and select the settings file. Under `General Settings`, change the `owner name` to something unique (like your name).



4 - eePe EEPROMN editor

Then go `Burn -> Write Memory to TX`.

Wait for the program to complete and confirm that there were no errors during the flashing process.

After flashing, remove the programmer and confirm that the new firmware is working correctly (you may get a few errors the first time you start it up, this is normal).

Transmitter setup

Note this section is simply a reference for manually changing the settings of the Transmitter to match the contents of the 9XR settings file in the previous section. If you have already uploaded the settings, you can skip this section.

Turn on the transmitter.

To go the global settings menu, press the LEFT button

Use UP/DOWN to cycle through the settings and LEFT/RIGHT to change the selected setting.

The settings that need to be changed:

- Owner Name = <your name>
- Beeper = NoKey
- Battery Warning = 9.9 V
- Splash Screen = OFF
- Splash Name = OFF
- Throttle Warning = OFF
- Switch Warning = OFF
- Mode = 1

Go to the calibration page:

And calibrate the sticks.

Press exit to get back to the main screen.

Press RIGHT to get to the model profiles. A number of different profiles (with their own settings) can be set up here. We need to create a profile for the quadcopter.

Press RIGHT again to get into the settings for MODEL01.

The settings for it should be:

- Name = Quadcopter

Go across to the MIXER page:

The channels should be setup as such:

- CH1 = 100% AIL
- CH2 = 100% ELE
- CH3 = 100% THR
- CH4 = 100% RUD
- CH5 = 61% HALF ID0
- -76% HALF GEA
- 10% HALF ID2
- 36% HALF ID1
- CH6 = 100% FULL TRN
- CH7 = 100% FULL ELE

Go across to the LIMITS page

Set CH2 to INV

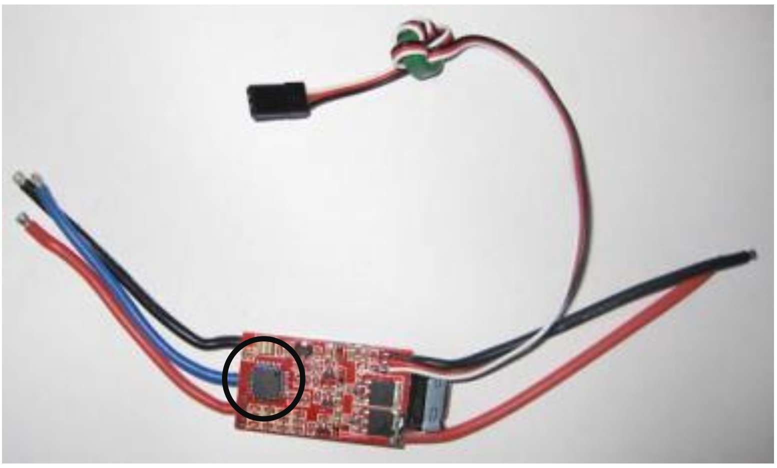
At this point, your transmitter is set up with its modes:

5 - Channel layout on the RC Transmitter

ESC Firmware Flashing

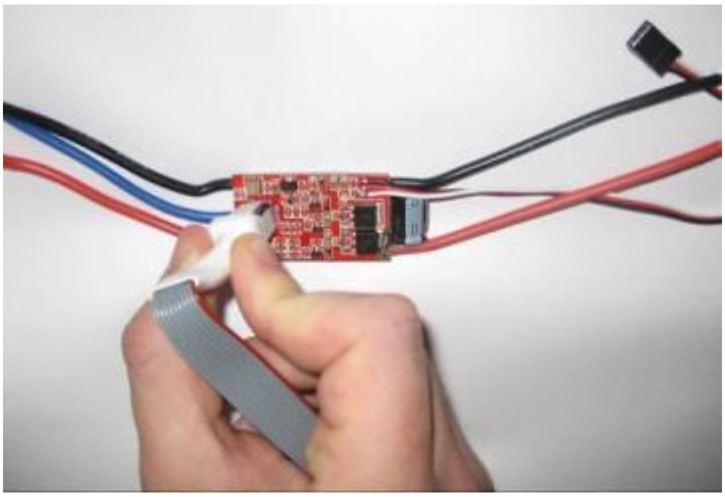
Note: For the purposes of this workshop, a computer will be setup with the flashing software ready to go.

Carefully cut off the heatshrink material, taking care not to damage the components underneath. Try cutting from the side with a knife



6 - Location of microcontroller on the ESC

For the purposes of this workshop we are using a dedicated firmware programmer device. Using this device, line it up over the microcontroller on the ESC (taking note of the correct orientation). Hook the programmer to a laptop and load the SimonK firmware.



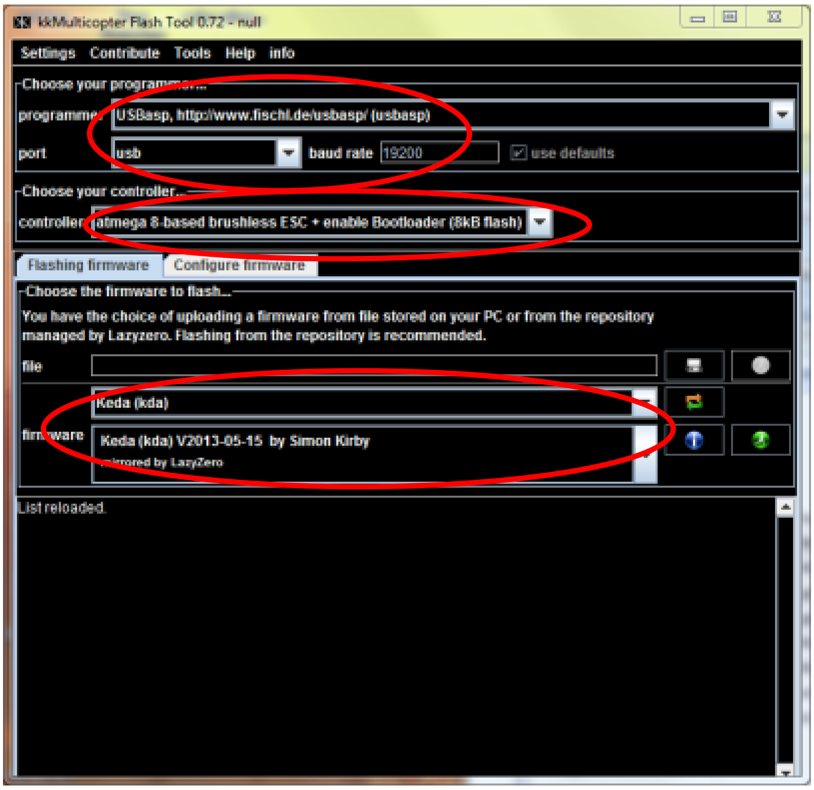
7 - Using the firmware flasher on the ESC

Use the “KKMulticopter Flashtool” from http://lazyzero.de/en/modellbau/kkmulticopterflashtool#download.

If you are running windows, ensure to install the USBASP driver first.

The black corner of the firmware flasher aligns to the pin 1 corner (the corner on the IC with the dot).

Use the “Keda (kda)” firmware variant with the “atmega 8-based brushless ESC + enable Bootloader”



8 - kkMulticopter Flash Tool setup for ESC flashing

Under Linux, grab the Github download of the SimonK firmware (https://github.com/sim-/tgy/downloads) and look for the “kda.hex” variant. This is the correct firmware for the Multistar 20A ESC’s. Use the following shell command to upload it:

```
avrdude -c usbasp -p m8 -U flash:w:kda.hex
```

(a `sudo` may be required on some systems)

Once programmed, apply (new) heatshrink to the ESC. Repeat for all 4 ESC’s.

Ignore any errors that say “warning: unable to sck period”. The ESC will still be programmed.

Simulation

The RC transmitter can be connected (via the supplied USB dongle) to your PC and used as a standard joystick. Combined with the (various) RC flight simulators, it can serve as a decent training module for new pilots and you can practice your flying skills in a safe environment.

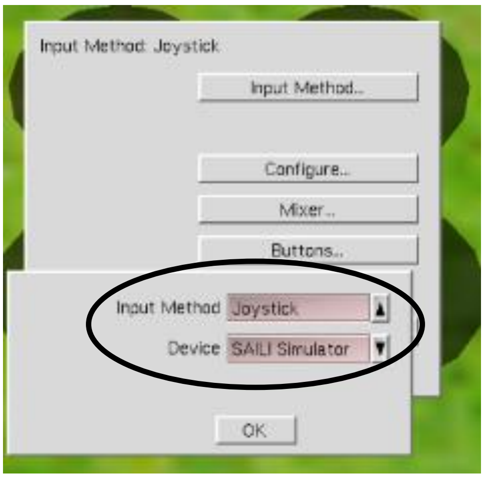
Simply connect the (audio stereo) cable to the rear of your RC transmitter this cable plugs into the dongle, which in turn connects to a USB port. The TX module must be removed (the box module on the back of the transmitter) and the power switch of the RC transmitter must be off.

There are a few options for RC Simulators. The most popular are:

- Realflight (http://www.realflight.com/) which is commercial
- CRRCSim (http://sourceforge.net/apps/mediawiki/crrcsim/index.php?title=Main_Page) which is open source.

Either will work well. In the case of this workshop we will be using CRRCSim.

Open CRRCSim, press the `esc` key to get the main menu. Go to `Options -> Controls -> Input Method` and enable the RC Controller:

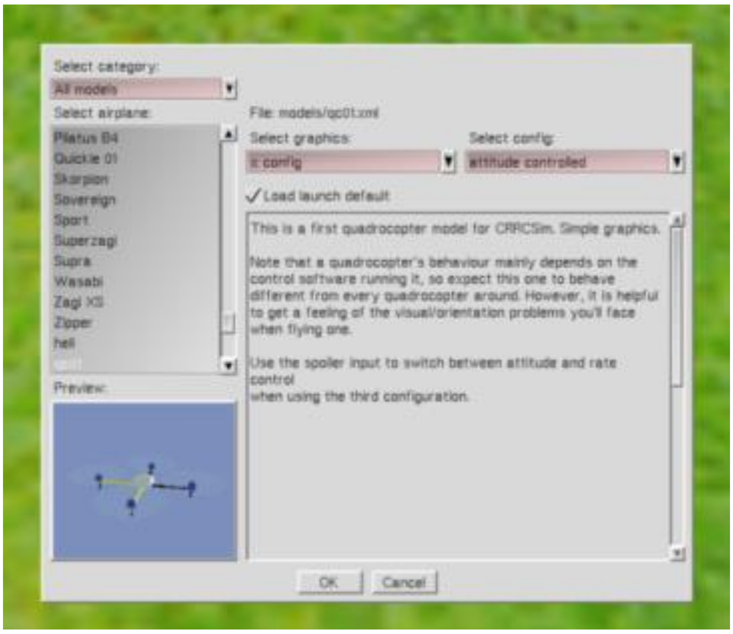


9 - CRRCSim Input Selection

Then go to `Options -> Controls -> Configure`. Select the correct channels for the sticks and calibrate the controller.

10 - CRRCSim calibration page

Go to `Options-> Aircraft` and ensure the x-config quadcopter (attitude controlled) is selected:

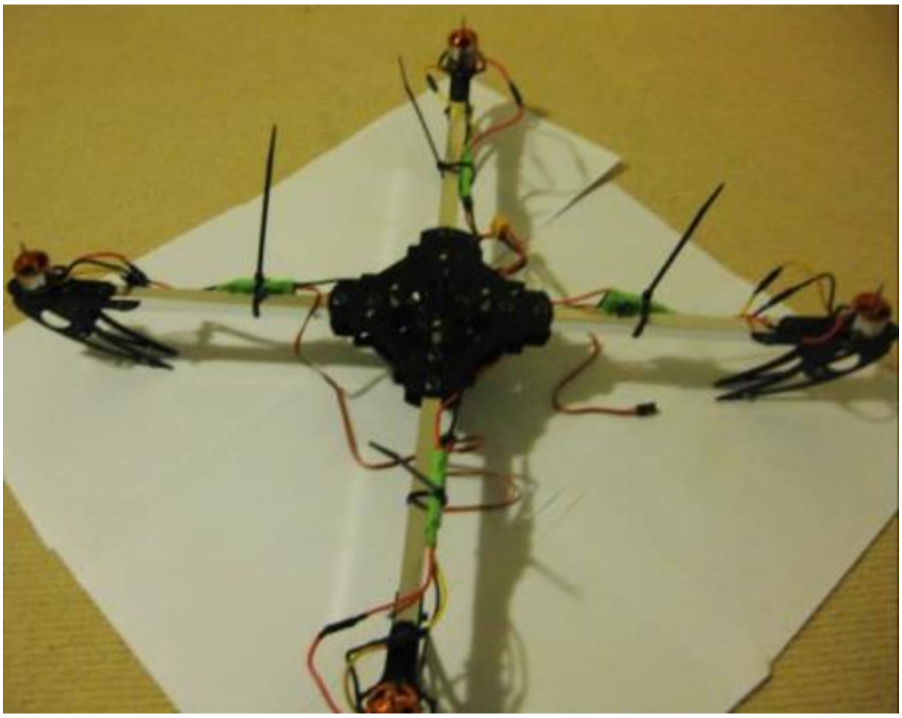


11 - CRRCSim model selection

Return to the simulation and test that the RC Transmitter controls the simulator correctly. Use this simulation to get a feel for the quadcopter's controls and movement.

Wiring up the system –Power train

Connect the ESC's up to the motors and thread the power distributor through between the hub plates and hook it up to the ESC's. Wire ties or tape can be used to affix the ESC's to the arms. They ESC's should be set back from the hub in order to reduce electrical interference to the Pixhawk.



The ESC <-> Motor connections should be left loose for the moment, as they may need to be changed later (to set the correct motor direction).

Propeller Balancing

Given how fast the propellers on the quadcopter spin, it is important that the propellers are balanced (equal mass on both sides). If they are not, the quadcopter may have a slight spin or drift when it is flying.

To balance them, you need a propeller balancer:

12 - Propeller balancer

Simply place the propeller in the rod and watch for movement. If one side of the propeller is clearly heavier, use some sandpaper to (gently) rub some mass off the top (the side with the embossed writing on it) of the propeller until it is balanced.

Repeat for all 4 propellers.

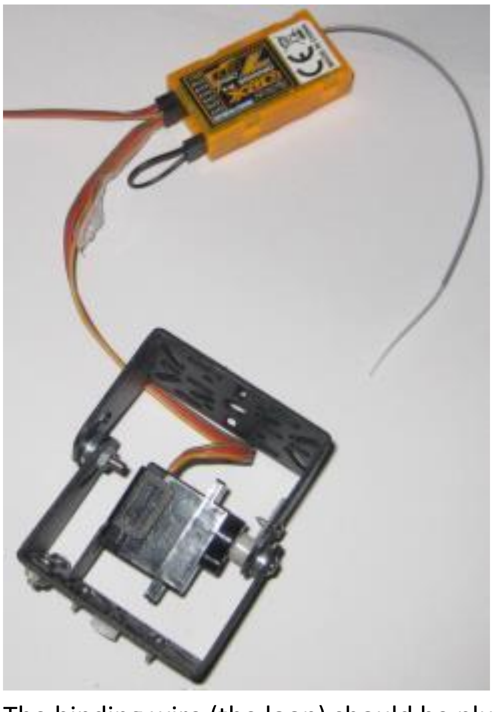
Radio binding (and servo test) and Failsafe

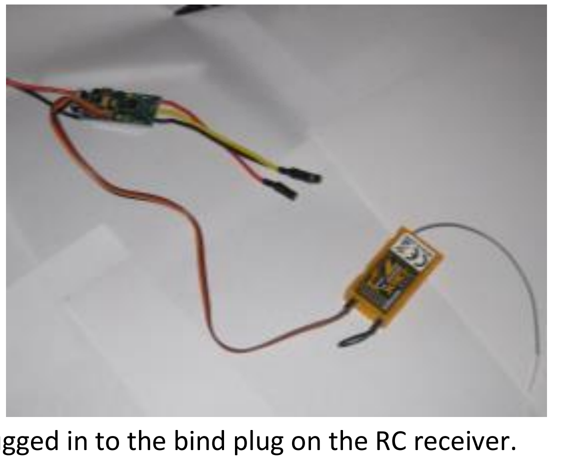
Binding is the action of syncing your transmitter and receiver. Due to the nature of the process, this can only be done one at a time.

For the servo cables: The brown cable goes to the top, red in the middle and orange on the bottom.

1. Plug in the TX Module to the Transmitter and connect the antenna. NOTE: Do not turn the TX Module on without the antenna connected – you could damage the TX Module.

2. Hook up one of the ESC's to the RC receiver to provide power. Attach a servo to channel 3.





3. The binding wire (the loop) should be plugged in to the bind plug on the RC receiver.
4. Power up the RC receiver by connecting the flight battery to the main plug on the quadcopter. The RC Receiver should be flashing at a fast rate
5. Whilst holding the bind button on the underside of the RC transmitter, switch it on.
6. Wait until the light on the RC receiver stops flashing. This take around 10 seconds
7. Turn off the RC transmitter, then the RC receiver.
8. Remove the bind plug from the RC receiver

Turn the RC receiver and transmitter back on. Confirm the binding works by twiddling the sticks and watching for movement from the servo.

Perform a range test on the Transmitter by walking 15-20m from the RC Receiver and pressing the bind button (low power mode on the RC Transmitter) and confirming the servo will still respond.

Evening 3

Safety

Current Australian Law and Regulations

Any UAV (no matter the size) being used for commercial operation requires a UAV Operator's Licence (expensive!). Otherwise, the UAV is covered under the Model Aircraft rules.

The Model Aircraft Rules specify:

- Keep your model aircraft away from populated areas and the immediate vicinity of others (i.e. more than 30 metres)
- Do not operate within 3nm (5.5 km) of an aerodrome without approval
- Do not operate in controlled airspace above 400ft (120m) without approval
- Do not operate in military prohibited or restricted areas without approval
- Do not fly in poor visibility, clouds or at night

The MAAA (Model Aircraft Association of Australia) are the recognised association for model aircraft pilots. They are mostly focussed on fixed wing aircraft, but can offer training (and airfields) for the use of members.

When flying at the local oval or in public places, it is **very** important to realise that if you crash into a person/car/house/etc, **you put yourself at risk of legal action and may have to pay compensation to the affected party**.

The general rules are:

- Do not buzz or fly near people without their OK.
- Consider your skill level, nearby obstructions and weather conditions in estimating your "safe zone".
- Don't rely on the APM to get you out of trouble.
- Only fly in situations where you are confident of manual recovery.
- Maintain a line of sight to the quadcopter.
- Watch out for trees!
- Windy conditions can really toss a quadcopter around. Take this into consideration when flying.
- If flying with a camera, be mindful of people's privacy.
- Don't fly over private property without the landowner's permission.

Note that, like most countries, Australia is in the process of reforming its laws and regulations regarding UAVs. Media coverage of people performing stupid or dangerous manoeuvres with UAVs will only encourage the authorities to make the regulations more stringent, or outright ban amateur UAVs.

13 – RC Aircraft crash (from www.theoildrum.com)

LIPO Batteries

The Lithium Polymer (LiPo) batteries that quadcopters use contain large amounts of energy in a small and light package.

Due to the nature of the chemistry in the batteries, they can catch on fire or explode if not maintained properly.

LiPo guidelines:

- When not in use, keep in a fire-resistant enclosure.
- Never charge a battery unattended.
- If a battery looks damaged or puffy - discharge it, cut off the leads and throw it away.
- Do not discharge the batteries below their minimum (use a battery alarm)
- Make sure you use a LiPo charger
- Keep a bucket of sand handy to put out a LiPo fire (do NOT use water to put out a LiPo fire).

14 - Results of a LiPo fire (from www.rctech.net)

Propellers

The propellers on a quadcopter spin at many 1000's RPM. They can cause serious injury if they were to hit a person.

It is thus very important that all people near an active quadcopter are aware of the quadcopters position at all times and a prepared to run/duck as needed to avoid getting hit by it.

The APM features an arm/disarm switch. When disarmed, the motors will not spin under any circumstances (barring a major software error in the APM). **The APM should only be armed at takeoff and immediately disarmed after landing.**

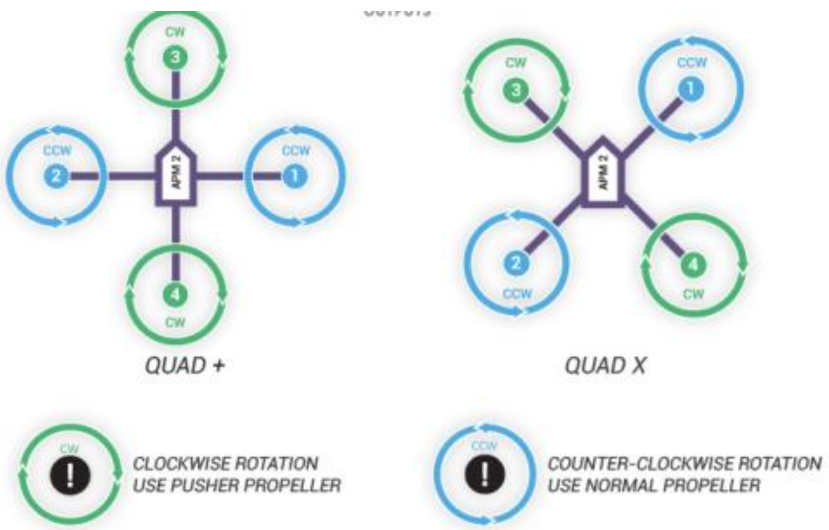
Radio Interference

Due to the large numbers of radios operating during the workshop, it is important that any radio transmitters (RC Transmitters, Ground station radio and Quadcopter radio) are turned off when not in use. This will lower the chance of harmful interference to other users.

Wiring up the system – Flight Electronics

There are a number if wire connections that need to be made. Note that the cables can be difficult to remove once plugged in – connect them when you are sure of placement. To make them easier to undo, consider trimming the two small tabs at the end of the wide blank face.

- GPS/Compass <-> Pixhawk – this uses 2 polarised cables. The GPS/Compass module should be as far away from the other radio transmitters as reasonably possible. The compass plugs into the "I2C" port on the Pixhawk.
- Telemetry radio <-> Pixhawk – this uses a polarised cable in the "Telem 1" port.
- RC Receiver -> Pixhawk (input) – use a single 3cm servo cable, going between the RC port on the Pixhawk and SBUS port on the RC Receiver.
- Pixhawks (output) -> ESC's – take careful note of which numbered motor goes to which Pixhawk output (see below). Look for the "Main Out" labels. The included stickers should be affixed to make the identification of ports easier. It's useful to use a marker to write the motor number and direction of motion on each arm too.

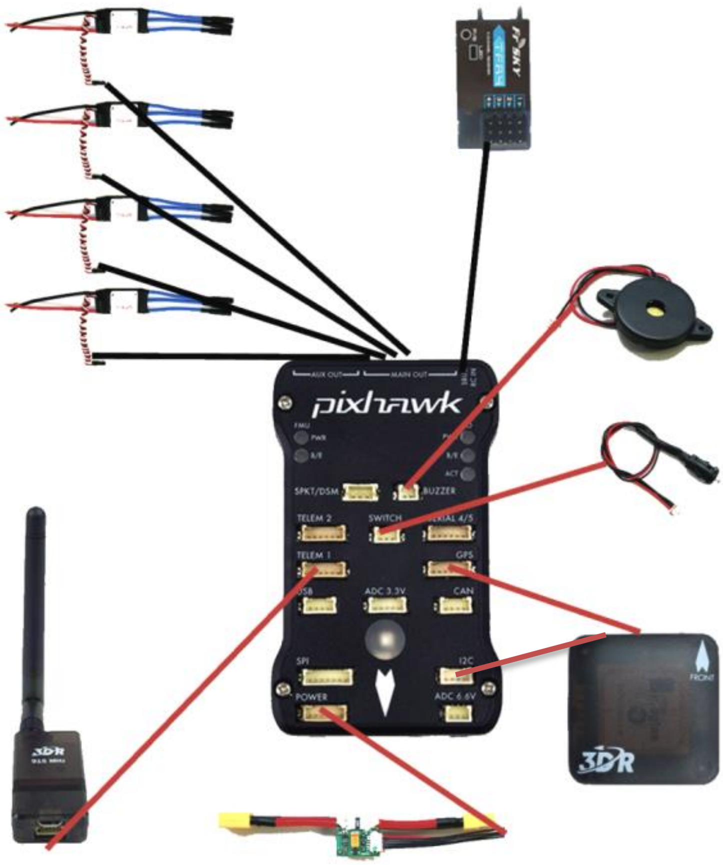


15 - Motor and propeller directions (from http://code.google.com/p/arducopter/wiki)

- Pixhawk <-> Power module – a single cable to the "Power" port on the Pixhawk. The power module itself sits between the battery and power distributor (hence the XT60 connectors). It provides stable power to the Pixhawk as well as voltage and current information.

- Pixhawk <-> Arm switch – a single cable to the “Switch” port on the Pixhawk. This button arms and disarms the quadcopter.
- Pixhawk <-> Buzzer – a single cable to the “Buzzer” port on the Pixhawk. This provides audio feedback of the status of the Pixhawk.

Putting all of these parts together gives:

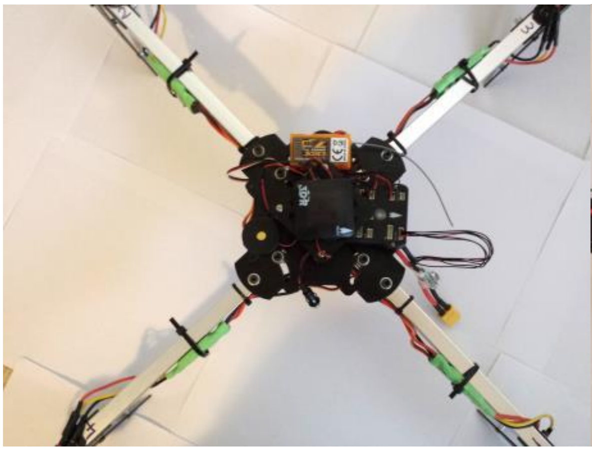


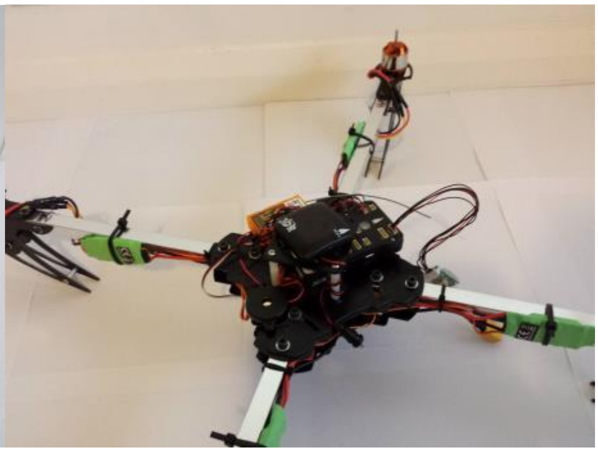
16 – Pixhawk system connections

The Pixhawk itself should be placed on the foam feet in order to reduce vibrations. The GPS/Compass module should be placed as far as practical from any high-power cables and ESC’s.

Note the forward arrows on the Pixhawk and GPS/Compass modules. In both cases they should be lined up to point towards the front of the quadcopter.

An example layout of the parts on the quadcopter:





In the layout above, the GPS/Compass module is place on the small top stand. A couple of standoffs raise the top stand high enough to accommodate the Pixhawk. The RC Receiver, buzzer, telemetry radio and switch are placed around the main base.

The telemetry radio and RC Receiver should be placed away from each other to reduce the chances of interference.

It is also worth colour-coding the landing legs to identify which direction is forwards on the quadcopter. This is useful in maintaining situational awareness during flight.

Connecting the ground station

Hook up the USB cable to the APM. This will provide enough to power the flight electronics, but not enough to activate the ESC's/motors.

The instructions for this section will show the setup via the 3 major ground station software programs – Mission Planner, MAVProxy and APM Planner 2.

The Android apps are limited in the amount of setup they can perform on the quadcopter, so they will not be covered in this section.

Firmware Loading

Mission Planner

Go to the *Initial Setup -> Install Firmware* tab.

Ensure the correct serial port is selected in the upper right of the window.

Click the image of the quadcopter to begin the upload process.

Follow the instructions to remove and re-plug the Pixhawk as required.

MAVProxy

As a first off, you will need to download a copy of the latest firmware:

http://firmware.diydrones.com/Copter/stable/PX4-quad/

The "ArduCopter-v2.px4" firmware is the one used on the Pixhawk.

A python script can be used to upload the firmware:

https://raw.github.com/diydrones/PX4Firmware/master/Tools/px_uploader.py

Then use the following command to upload the firmware:

```
python px_uploader.py --port /dev/ttyACM0
```

where */dev/ttyAM0* is the name of the serial port that the Pixhawk is using.

APM Planner 2

Go to the *Initial Setup -> Install Firmware* tab.

Ensure the correct serial port is selected in the drop-down box next to the *Stable* button.

Select the *Stable* button to use the stable branch of the firmware.

Click the image of the quadcopter to begin the upload process.

Follow the instructions to remove and re-plug the Pixhawk as required.

After the firmware load, the Pixhawk will play a musical tone to indicate that it has booted correctly. If it plays a series of tones and then three beeps, disconnect the USB and press down the safety switch. Reconnect the USB and wait for a series of tones followed by two beeps. This is sometimes required if the failsafe board requires it's firmware to be updated. The failsafe board firmware is included in the main Pixhawk firmware package and will be upgraded if necessary

Telemetry Radio Setup

Mission Planner

Make sure you are not connected to the APM.

Go to `Initial Setup -> 3DR Radio`:

Change the `NET ID` number to a random number between 0 and 65536.

Check the below settings list to ensure all your settings are correct.

MAVProxy

Go to a serial console (telnet, etc) and connect to the radio

Type `+++` to enter command mode

`ATI5` to show settings (local)

`RTI5` to show settings (remote radio)

`ATS4=10` to change setting **4** to value **10** (local), for example.

`RTS4=10` to change setting **4** to value **10** (remote), for example.

`AT&W` to save settings to EEPROM (local)

`RT&W` to save settings to EEPROM (remote)

`ATO` to exit command mode

Check the below settings list to ensure all your settings are correct.

APM Planner 2

Make sure you are not connected to the APM.

Go to `Initial Setup -> 3DR Radio`:

Change the `NET ID` number to a random number between 0 and 65536.

Check the below settings list to ensure all your settings are correct.

The correct settings for use in Australia should be:

- Frequency – 91500 to 928000
- # of (hopping) channels >= 20
- Tx Power 20 (dBm)
- Net ID = random unique number. Use this to differentiate yourself from all the other transmitter/receiver pairs. **Anyone with the same NET ID as you can read/command your telemetry data.**

Make sure the local and remote radios are both changed before pressing the “Save settings” button. The radios will need to be rebooted before the new settings take effect. Connect to the APM via the radios and confirm that you’re getting the telemetry.

Telemetry Radio Connection

Mission Planner

Watch for roll/pitch changes

Connect here. Baud rate is 115200 for USB cable, 57600 through radio

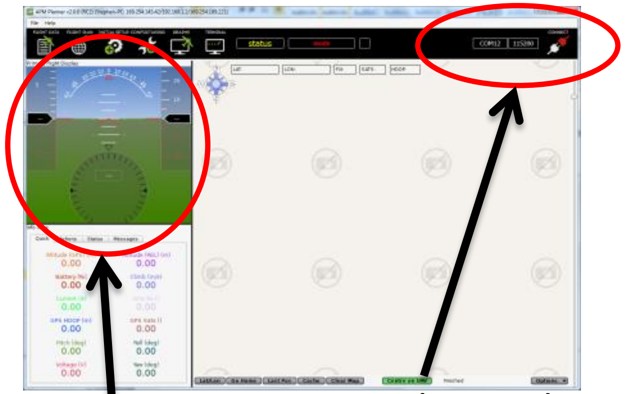
MAVProxy

To connect via MAVProxy:

mavproxy.py --baud=57600 --console

It will auto-detect the USB-Serial port (assuming you only have one connected). Otherwise use the *--master=<port>* argument

APM Planner 2



Watch for roll/pitch changes

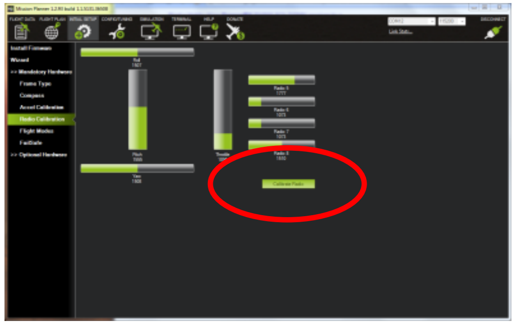
Connect here. Baud rate is 115200 for USB cable, 57600 through radio

Rotate the quadcopter and check the roll/pitch values look sensible. Go outside and confirm a solid GPS lock.

RC Radio Calibration

Mission Planner

Connect to the Pixhawk and go to `Initial Setup -> Mandatory Hardware -> Radio Calibration`:



Press the `Calibrate Radio` button and follow the prompts.

MAVProxy

In a console, connect to the Pixhawk in setup mode:

`mavproxy.py --baud=57600 –console --setup`

To start the calibration process:

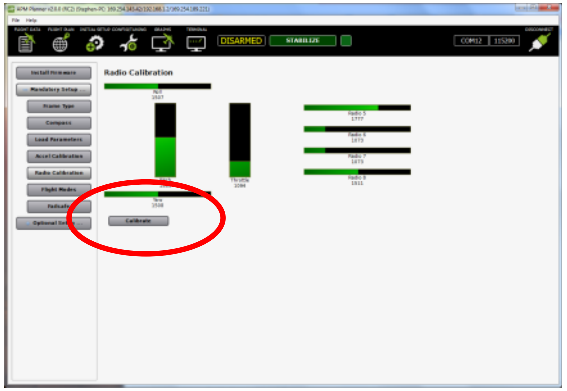
`ArduCopterMega]` *`setup`*

`Setup]` *`radio`*

Follow the prompts.

APM Planner 2

Connect to the Pixhawk and go to `Initial Setup -> Mandatory Hardware -> Radio Calibration`:



Press the *Calibrate Radio* button and follow the prompts.

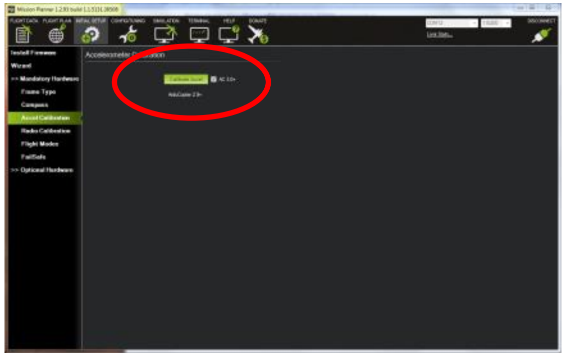
Waggle the RC sticks and buttons to confirm they all work.

Go through the above process so the Pixhawk knows the min/max extents of your RC controller.

Accelerometer Calibration

Mission Planner

Connect to the Pixhawk and go to *Hardware -> Mandatory Hardware -> Accel Calibration*:



Ensure the AC 3.0+ checkbox is ticked and press the *Calibrate Accel* button and follow the prompts.

MAVProxy

In a console (if not already connected), connect to the Pixhawk in setup mode:

mavproxy.py --baud=57600 –console --setup

To start the calibration process:

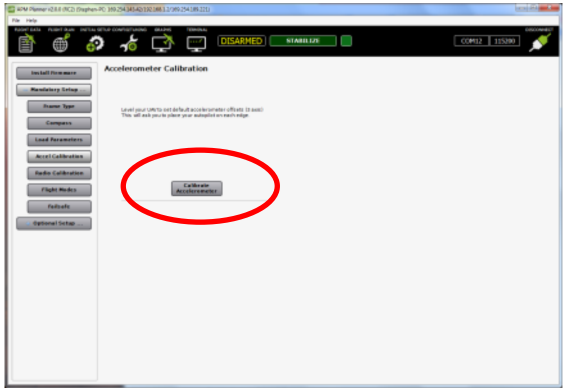
ArduCopterMega] *setup*

Setup] *accel*

Follow the prompts.

APM Planner 2

Connect to the Pixhawk and go to *Hardware -> Mandatory Hardware -> Accel Calibration*:



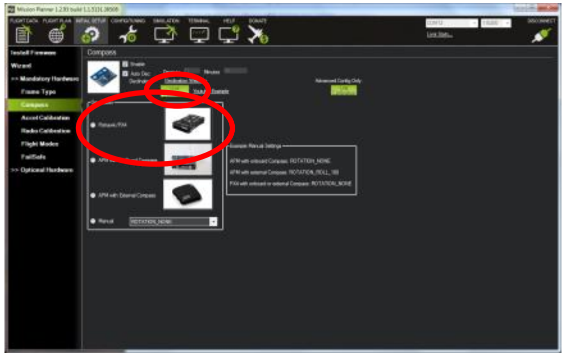
Press the *Calibrate Accel* button and follow the prompts.

This calibration enables the Pixhawk to account for not being perfectly flat mounted on the quadcopter and for any individual variations in the accelerometer measurements.

Compass Calibration

Mission Planner

Connect to the Pixhawk and go to `Initial Setup -> Mandatory Hardware -> Compass`:



In the `Orientation` option, select `Pixhawk/PX4`

Calibrate the compass by pressing the `Live Calibration` button and following the procedure.

MAVProxy

In a console (if not already connected), connect to the Pixhawk in setup mode:

```
mavproxy.py --baud=57600 –
console --setup
```

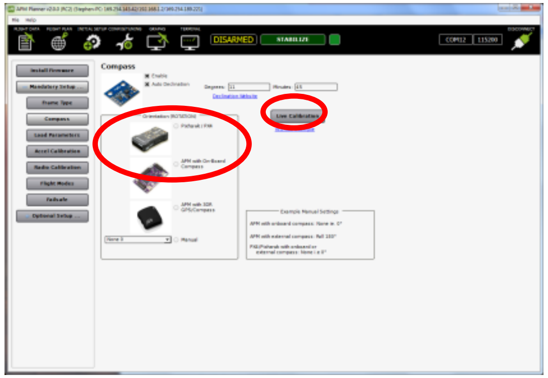
To start the calibration process:

```
ArduCopterMega] setup

Setup] set COMPASS_DEC 12.19

Setup] set COMPASS_ORIENT 8

Setup] set AUTODEC 0

Setup] compassmot
```

Follow the prompts

APM Planner 2

Connect to the Pixhawk and go to `Initial Setup -> Mandatory Hardware -> Compass`:



In the `Orientation` option, select `Pixhawk/PX4`

Calibrate the compass by pressing the `Live Calibration` button and following the procedure.

This calibration enables the compass to account for magnetic declination in Canberra (the 12.19), the orientation of the compass on the quadcopter (it's upside-down on the GPS/Compass module) and any local metal sources on the quadcopter (calibration/compassmot).

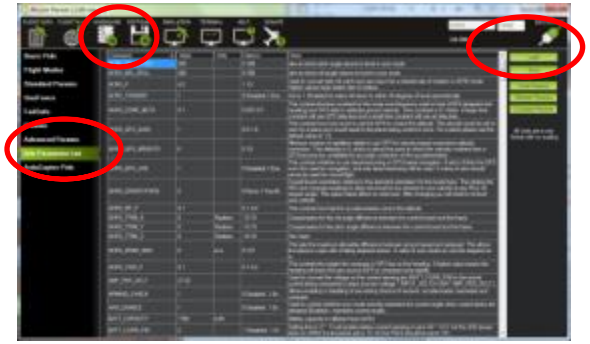
Loading Parameters

The settings for the Pixhawk are known as the parameters (or params for short). They can be exported/imported the Pixhawk via a simple text file.

To make things easier for everyone, we will be loading a set of known parameters. They are available from https://canberrauav.readthedocs.org/en/latest/quadcopter-workshops/quadcopter-workshop-2.html

Mission Planner

Connect to the Pixhawk and go to *`Software -> Adv Parameter List`*:



Press `Load` to open the open the parameters file.

Press *`Write`* to write the parameters to the Pixhawk.

MAVProxy

In a console (if not already connected), connect to the Pixhawk (note **not** in setup mode):

`mavproxy.py --baud=57600 --console`

`param load MHV.param`

APM Planner 2

Connect to the Pixhawk and go to *`Initial Setup -> Mandatory Hardware -> Load Parameters`*:

Click on the `Load file` and then `Download` to load the parameters onto the quadcopter.

Motor and ESC tests and calibration

Ensure the propellers are not fitted to the quadcopter for this section

First the ESC's need to be calibrated to match the signal levels given out by the RC transmitter. It is done by:

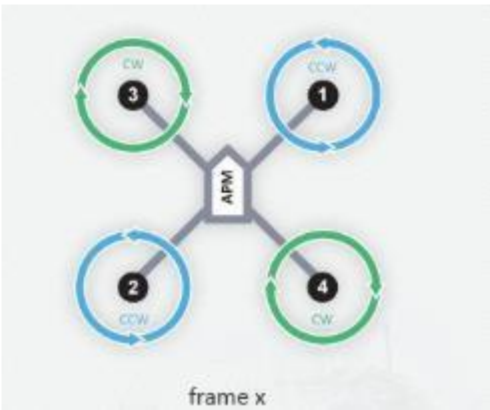
1. Disconnect USB and battery. The entire system must not have power
2. Turn your RC transmitter on and put the radio throttle stick at maximum.
3. Connect the battery to power the quadcopter.
4. Leaving the throttle stick at full, disconnect the battery and reconnect it again to reboot the APM and power the quadcopter. The APM will now pass the radio signal directly to the ESCs, triggering calibration. The LED will cycle between red, green and blue.
5. Press the safety switch for 5 seconds to arm the quadcopter.
6. Wait for the ESC's to beep three times.
7. Put the throttle to 0 and wait for a single beep
8. Move the throttle up a little to confirm all ESCs are armed and the motors are working in sync. It is important that all motors start/stop at the same time.
9. Your ESC's are now calibrated.

Next, the motor spin direction needs to be checked.

Hook up the battery to the APM and power up the quadcopter.

Push rudder (left stick) right for 4 seconds to ARM the quadcopter. Increase the throttle to 1/5 and confirm that all the motors spin.

Carefully put your finger on the motor (or use your eye) to determine which way the motors are spinning. It should match this:



17 - Motor and propeller directions (from http://code.google.com/p/arducopter/wiki)

If they do not, reduce throttle to 0 and DISARM push rudder to the left for 4 seconds. Confirm the DISARM by pushing the throttle up – the motors should not engage. Swap (any) two power leads (ESC<->motor) of the affected motors to reverse spin direction.

ARM the quadcopter and throttle up again to confirm the motors now spin correctly.

Build Checklist

Check that all of the following activities have been completed before flight

- Transmitter settings loaded and calibrate
- Receiver bound to transmitter
- Telemetry radio NET_ID changed
- ESC's calibrated
- All propellers give an upward force when spun up
- Pixhawk settings loaded
- Pixhawk accelerometers, RC channels calibrated
- Pixhawk compass calibrated
- Confirm GPS lock

Test Flights

Activate the ground station software and connect to the Pixhawk. Confirm that telemetry is downloading.

Add the propellers and battery to the quadcopter and give it ~20 seconds to boot. If you have your laptop handy, launch your ground station software and connect to the quadcopter via the telemetry radio links.

ARM the quadcopter by:

1. Pressing and holding the safety switch of the quadcopter until the ESC's beep.
2. Tilting the left stick to the right for 4 seconds.

The light on the Pixhawk go solid green and a long low tone will play. The ground station will also show that the quadcopter is armed.

DISARM the quadcopter via the reverse process.

For safety reasons, the quadcopter will not ARM if the throttle is not at zero of if it does not have an accurate (<2m HDOP) GPS lock.

Raise the throttle to 25%. Gradually raise is further until the quadcopter takes off. Once it is 1m off the ground, slowly decrease the throttle for a gentle landing.

If the trims on the quadcopter are slightly off it will slip to one side.

Repeat the small hops a number of times until you are familiar with the throttle levels needed for a safe takeoff and landing.

Start making the hops a bit longer and use the throttle to maintain a safe altitude.

Evening 4

Safety

Due to the large number of beginner flyers, keep a lookout for any low flying quadcopters!

When about to takeoff, if it looks too crowded, wait until someone else has finished flying.

Startup

Power up the ground station, RC transmitter and quadcopter. Confirm the numbers on the ground station look good.

Longer flights

Takeoff and fly up to 3m altitude.

Use the sticks to navigate the quadcopter around a simple box pattern.

At any time you lose control, switch the quadcopter to LAND mode (top button on right edge)

LOITER

LOITER mode will keep the quadcopter at the current lat/long/altitude (within a few metres) and will compensate for any wind.

Test this mode out and use your sticks to change the lat/long/altitude setpoint.

RTL

Fly the quadcopter around for a bit and switch the Channel 7 (which is mapped to RTL in the provided params file) to on. The quadcopter will then fly to 5m altitude, fly to the point at which the quadcopter was ARMed and land. This mode is very useful if you ever lose control of your quadcopter.

ALT_HOLD

This mode will attempt to keep the quadcopter at the current altitude, whilst giving full roll/pitch/yaw control to the pilot.

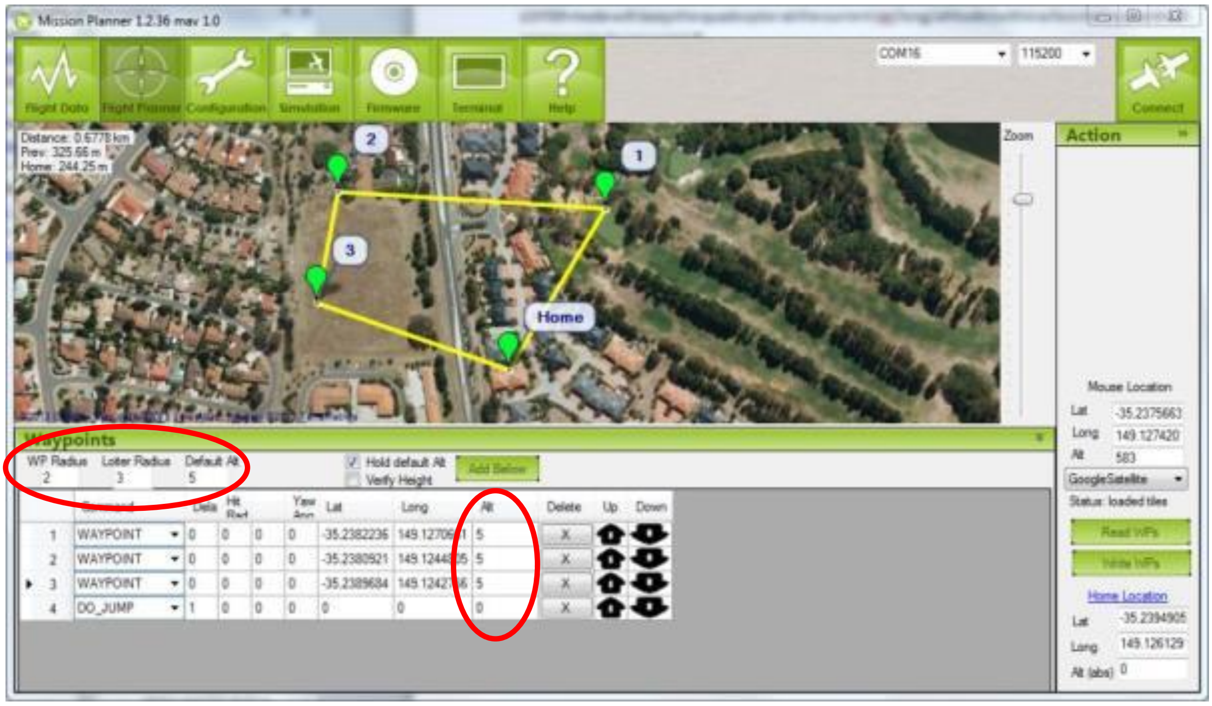
Only switch to ALT_HOLD mode when you are level at the altitude you want to maintain. It may take up to 30 seconds for ALT_HOLD to come into full effect.

AUTO

This mode is fully automatic, with no input needed from the pilot.

AUTO mode requires a *mission* (set of waypoints) in the Pixhawk's memory in order to function.

This is a simple point-and-click affair on the APM Mission planner (left click on the map to add points, right click for advanced options):



In this case the quadcopter will fly to waypoints 1, 2 and then 3. The DO_JUMP waypoint at the end tells the Pixhawk to go to waypoint 1. Thus it will repeat the set of waypoints until told otherwise. Despite what the above image indicates, it will NOT go to the home point.

There are numerous waypoint types to tell the APM to change speed/altitude, loiter at a position for a period of time, return to land and others.

Of particular note are the altitude settings (red circles). Double check them before your write the waypoints to the APM. The radius settings tell the APM how close it needs to be to a waypoint to consider it “reached”.

Note that all altitudes are relative to the ARM point. Be careful when setting waypoints around hills and other obstacles. Leave plenty of clearance to ensure the quadcopter doesn’t fly into any of these objects.

References

Arducopter Manual

http://copter.ardupilot.com/

APM source code:

https://github.com/diydrones/ardupilot

DIYDrones community site

www.diydrones.com

MHV Quadcopter repository

https://canberrauav.readthedocs.org/en/latest/quadcopter-workshops/quadcopter-workshop-2.html

ER9X firmware:

http://code.google.com/p/er9x/

ESC Firmware:

https://github.com/sim-/tgy