Montgomery County Community College

Basic quad information and troubleshooting –R4 Final

Research Document – G04

EGR299 Students and Engineering Faculty

2012

Table of Contents

THIS PAGE BLANK

Introduction

Welcome to the basic information and troubleshooting document!

This document consists of 6 chapters:

- ***The Quad*** – frame, motors, ESC, BEC, propellers, etc.
- ***The Communications*** – radio receiver and telemetry module
- ***The AutoPilot*** – APM2, PIDs, LEDs and general wiki information
- ***The Mission Planner*** – ground control program
- ***Troubleshooting*** – troubleshooting the DJI F450 and the APM2
- ***Glossary*** – glossary of terms

Please note that the autopilot and the software to control it are open source. The code that runs it is community-created and tested. It has been loaded and successfully run on many different platforms by many different people. This does not mean that the autopilot and its software are fool-proof. Read all literature pertaining to the quad and its software before flying. While flying always use caution and follow all safety procedures. Also note that this document is written for the ArduCopter 2.7 firmware and the APM Mission Planner 1.2.1. Older versions of the firmware and of the Mission Planner are not supported. Newer versions may correct some bugs and add/or new ones, update the ArduCopter and/or the APM Mission Planner at your own risk.

The MC3 students and faculty recognize and show appreciation the DIY Drones[1] community for the pictures and the wealth of information found on their blogs and forums. Special thanks to Mr. Sean Garnett from Skymax Engineering for his patience and dedication teaching us how to fly a quadrotor!

Credits

The EGR299 class students

Aaron Wilz
Daniel Rowe
Jacob Simon
Jason Cassel
Kevin Healy
Michael Marino
Michelle Gavan
Scot Kantner
William Fellmeth

The engineering faculty

Andrew Ippolito
Bill Brownlowe
Jean-Jacques Reymond

[1] http://diydrones.com/

THIS PAGE BLANK

Chapter 1 – The Quad

A quad, short for quadrotor, is an aerial vehicle. It produces lift not through fixed wings, but through four rotating propellers, two spinning clockwise and two spinning counterclockwise. The quad maneuvers by changing the rotational speed of these propellers. We are using propellers from DJI Innovations (DJI). They are 10 inches long and will bend upon impact instead of breaking. It is possible to bend the propellers back into shape but this will reduce the efficiency of the quad; we do not suggest this practice.

In our case the propellers are powered by 920KV[2] motors from DJI. Each motor is connected to a 30A Opto 11.1-14.8V LiPo Electronic Speed Control (ESC). Each ESC controls how much voltage is applied across its motor. In this way, the ESC tells its motor how fast to spin. The ESCs are soldered to the baseplate's battery connection via their red and black power wires and are connected to the ArduPilot Mega version 2 (APM2) via their three colored wires orange, red, and brown. The ESCs obtain power from the battery through the red and black power wires and obtain instructions, e.g. how fast to spin the motors, from the three colored wires.

The APM2 sends instructions to the ESCs based on the input from its various sensors[3] and the mode which it is in. For example, in "Stabilize Mode" in the APM2 tries to stabilize the quad by reading input from various sensors. In "Acro Mode" the APM2 does not try to stabilize the quad and as such does not read input from as many sensors; it runs mostly on input from the gyroscope.

The APM2 is powered, not from the ESCs, but from a Battery Elimination Circuit (BEC). The BEC is soldered to the base plate, like the ESCs. It is then connected to the APM2. The BEC takes the 11.1V coming from the battery and steps it down to the 5V that the APM2 can use. The APM2 than distributes these 5V to the telemetry module and the radio receiver.

All of these electronics are supported in a frame from DJI[4].

When correctly connected and powered each ESC should beep a number of times, from 0 to 6 times, and then play a tune. If the ESCs keep beeping it means that they are not calibrated.

[2] 920 rotations per volt applied across the motor.

[3] See Chapter 3 - AutoPilot for more detail.

[4] The making of vertical takeoff unmanned air vehicle (VTUAV) quadrotors.

To calibrate an ESC

1. With NO BATTERY and NO PROPELLERS CONNECTED, plug the three colored wire plug of the ESC you want to calibrate into the throttle channel of your RC receiver.
2. Turn on your transmitter and push the throttle stick to full up.
3. Plug the ESC into the LiPo battery.
4. You will hear a musical note and then 2 beeps. After the 2 beeps drop the throttle to full down. You will then hear 3 beeps for the 3-cell LiPo battery and then a single longer beep indicating the end points have been set and the ESC is calibrated and ready to fly. You may now disconnect your LiPo battery.

Repeat this procedure for the three remaining ESCs.

If it appears that the ESCs did not calibrate then the throttle channel on the transmitter might need to be reversed. After manual setting of all ESCs reconnect your receiver and ESCs and ensure your transmitter's mode select switch is set to the "Stabilize Mode" position. On power up, after a few beeps, the ESCs should play a tune and you should be able to arm the motors by holding the throttle stick down and to the right for about 5 seconds.

Chapter 2 – The Communications

The quad has three embarked communications modules on it: wireless telemetry module, radio receiver module, and mission computer module. This section details the first two modules: the mission computer is detailed in a separate document. The quad uses 433MHz modules from 3DR[5] to send and receive wireless telemetry. The embarked radio control module is a Spektrum AR8000[6] 2.4GHz 8-channel receiver.

Wireless Telemetry

The telemetry module is used to wirelessly send data about the quad to the APM Mission Planner as well as to receive commands, see Chapter 4 for more information on this data. To accomplish this, two modules are shipped; one on the quad and the other on the Acer tablet. Both of the modules are identical; they both transmit at 57600bps and share the same network ID. It is possible to edit the telemetry parameters through the APM Mission Planner but we do not suggest it because editing them incorrectly will cause the modules to stop working.

The telemetry antenna on the quad should be pointing down while the antenna on the field computer should be pointing up. This limits the range of the telemetry while the quad is on the ground but it extends the range while the quad is in the air. The antenna on the quad should also be placed away from the ESCs and other transmitting antennas as much as possible. Do this to decrease the interference and increase the range of the wireless telemetry.

Radio Receiver

The radio receiver is used to manually control the quad. The commands from the Spektrum radio transmitter are received by the Spektrum radio receiver and sent to the APM2 to control the direction of the quad and a number of parameters. The parameter you will most likely be changing is the "Flight Mode", the other changeable parameters are detailed in Chapter 4. This is changed by flicking the 3-position switch on the radio transmitter; this is how you will change the quad from manual control "Stabilize Mode" to automatic control "Auto Mode".

The radio antenna, like the wireless telemetry should be as far from interference as possible. The Spektrum receivers are mounted on the quads in there correct positions and should not have to be changed. Read the manual before changing anything about the Spektrum receivers.

Notes

The telemetry transmitting at 433MHz and the radio control transmitting and receiving at 2.4GHz can face serious interference from outside sources that transmit on their respective frequencies. When flying away from centers of population and large antennas this should not be a problem. If you are flying where there is a source transmitting at 2.4GHz or 433MHz, fly as far from the sources as possible because this interference could cause the quad to malfunction and crash.

---

[5] http://store.diydrones.com/3DR_RadioTelemetry_Kit_433Mhz_p/kt-telemetry-3dr433.htm

[6] http://www.spektrumrc.com/Products/Default.aspx?ProdID=SPMAR8000

THIS PAGE BLANK

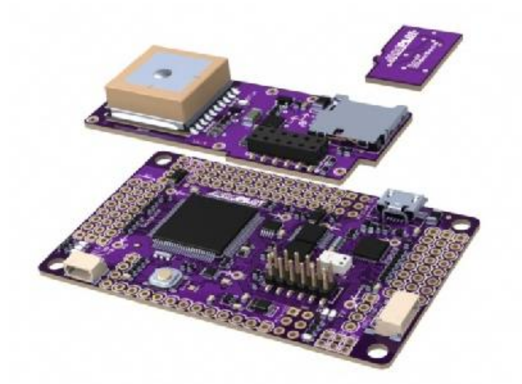
Chapter 3 – The AutoPilot

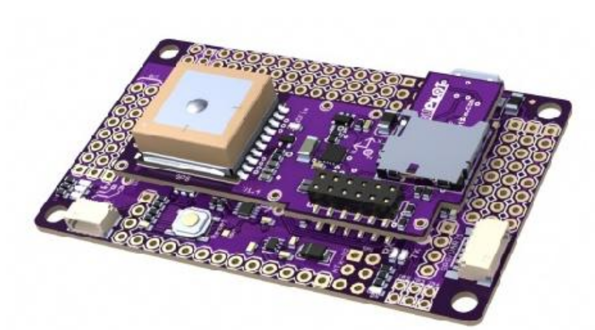
The autopilot on the quad is called the ArduPilot Mega 2 (APM2). It is manufactured by 3D Robotics and was created by 3D Robotics #DR) in conjunction with the DIY Drones[7] community

Main Features

- Three processors--a triple-core autopilot
- All new state-of-the-art sensors; the first autopilot to use the Invensense MPU-6000
- Smaller, lighter with GPS, magnetometer and dataflash included
- World’s only universal autopilot as the same hardware can autonomously control planes and multiquads
- Best-of-breed mission planning and two-way telemetry, and advanced scripting with Python for robot acrobatics
- Twice as much dataflash memory with microSD card slot
- When using the internal sensor fusion processor of the MPU-6000, more than half of the Atmega2560 processing capacity is free for new advanced features
- Native USB, with all new PPM encoder software[8]

The pictures below show the layout of the two boards that make up the APM2. The smaller board is a daughterboard that is placed on top of the larger motherboard, the microSD card is inserted in the daughterboard.





[7] http://store.diydrones.com/

[8] The PPM encoder is a bit of software you may have never heard about, mostly because it works really well in the background converting RC signals into something it's easier for a microprocessor to use. It's the software running on APM's Atmega328 secondary processor, and it also serves as the board's failsafe.

Sensors

The big advance in APM 2.0 is the introduction of the Invensense MPU-6000 sensors, which have an internal Digital Motion Processor (DMP) that does advanced sensor fusion.

DIY Drones upgraded the barometric pressure sensor to the MEAS MS5011 which has a resolution of 10cm and should give the quad best-of-class altitude hold capability.

Micro daughterboard

But why does it have a small daughterboard with a microSD card slot, GPS and the magnetometer on top? DIY Drones was very concerned about leaving the GPS and the compass stacked on the main board because what would happen to the compass if the board is placed near a big electromagnetic fields like a brushless motor? What would happen if the board is inside a carbon fiber frame and the GPS reception is blocked?

To solve these problems DIY Drones developed a small shield that can be mounted inside the boundaries of the pins and has special connectors to keep a very low profile of the system. But this shield is optional, so you can still attach your old GPS by using the standard APM1 GPS connector or the classic compass port. DIY Drones has created a special I2C port, similar to the GPS that allows you to attach an official APM2 Compass board by just plugging it. The microSD card slot is there because there is no other place with easy access. In the other hand the daughter board will come in four flavors: GPS+MAG+SD, GPS+SD, MAG+SD and SD. The supplied APM2s all come with the GPS+MAG+SD daughterboard soldered to the motherboard. To add an external GPS see Appendix 2.

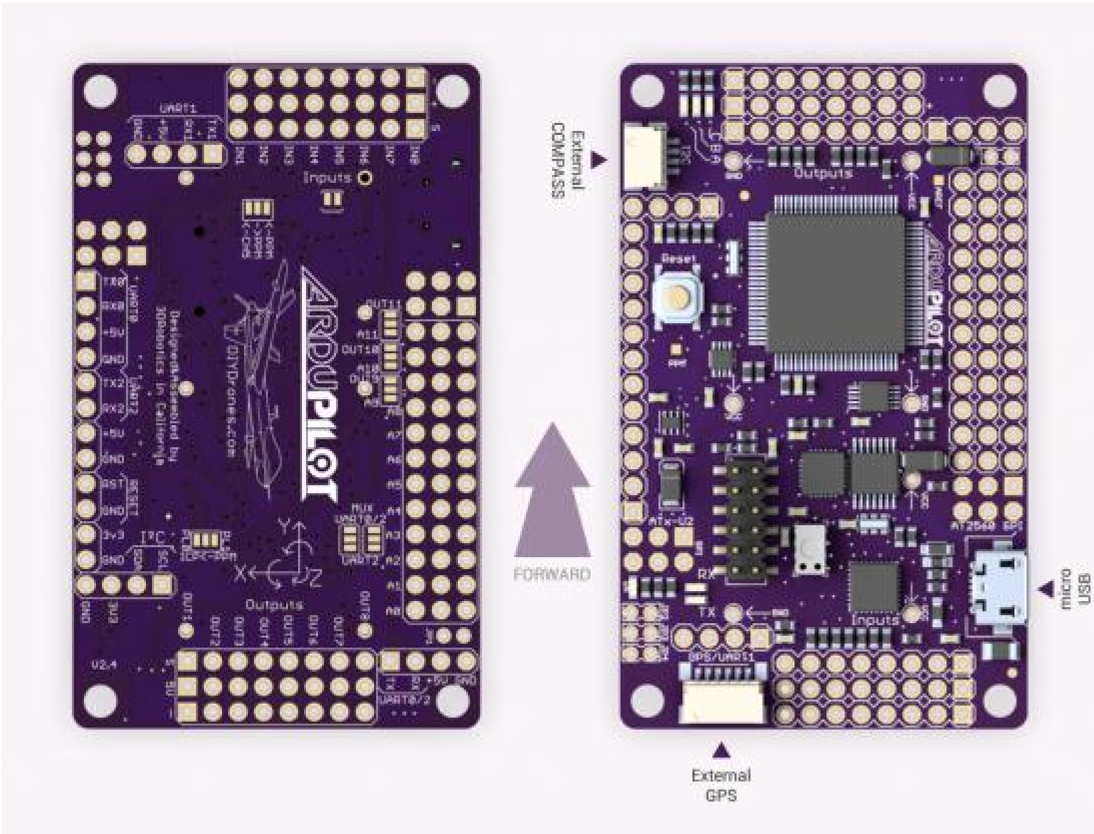
MicroSD card

The microSD card slot can read regular SD cards, but for the moment they are not in use because writing regular FAT tables is very slow and could screw up the main loop refresh rate since DIY Drones is not using a RTOS. Writing text files to a regular SD card from the APM2 is outside the scope of this document.

New PPM encoder and USB interface

Along with the Atmega2560, there is an Atmega32-U2 that works as the USB (FTDI) serial programmer (Arduino-compatible) and PPM Encoder. Best of all, you can update the Atmega32-U2 firmware without buying a SPI programmer and you can easily update via USB.

The Atmega32-U2 also features something called “Serial0 Auto Switch.” This function automatically toggles the serial port 0 from the Atmega2560 to the USB Serial programmer and the modem/OSD port. When you are about to upload a new code through the mission planner or the “Arduino” Atmega32-U2, it will auto-route the Serial0 to the USB COM port and load the code. When done, it will automatically switch it back to the modem or OSD port.

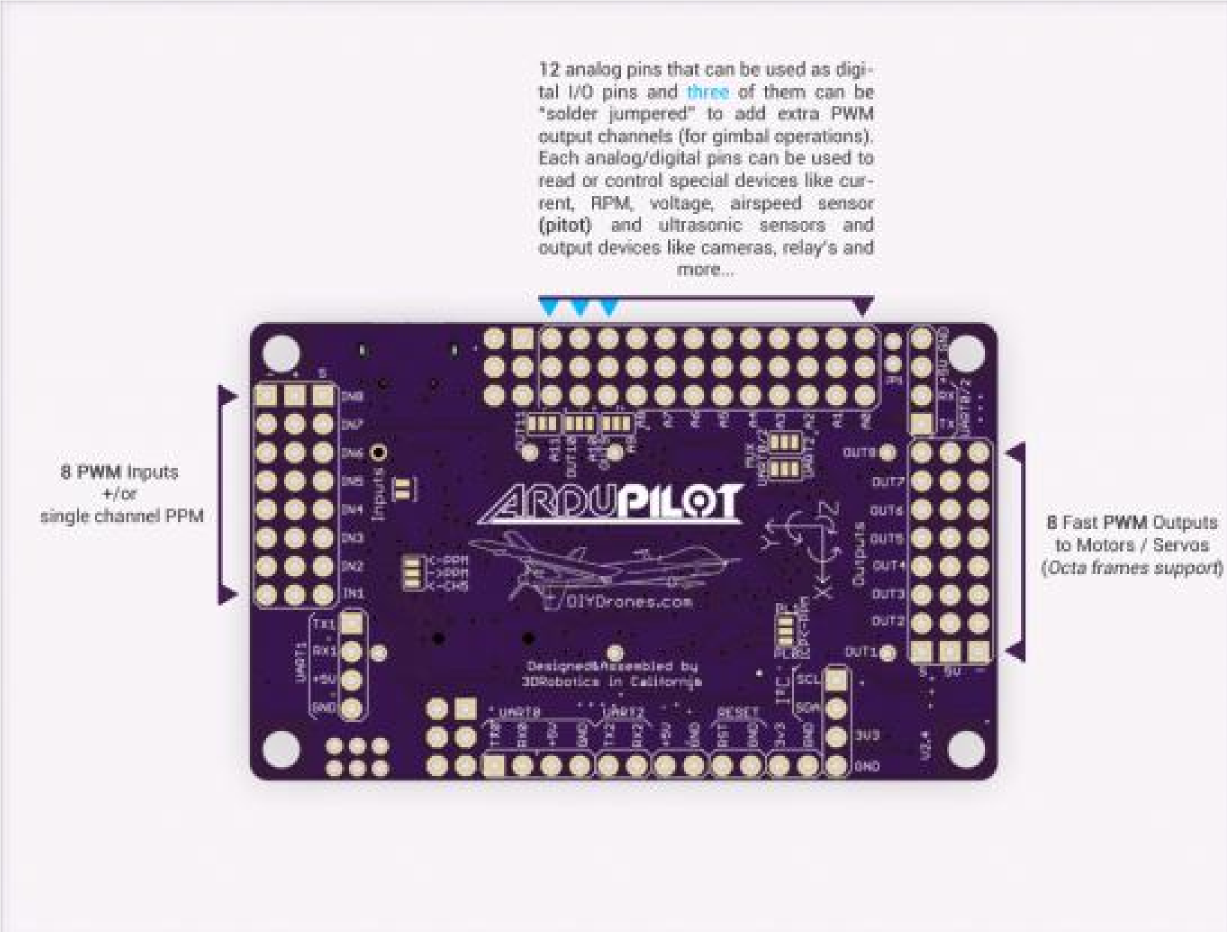


More I/O

The APM2 is also packed with 12 analog pins that can be used as digital I/O pins and 3 of them can be "solder jumpered" to add extra PWM output channels for example, gimbal 3-axis camera operations. Each analog/digital pins can be used to read or control special devices like current, RPM, voltage and ultrasonic sensors, and output devices like cameras and relays.

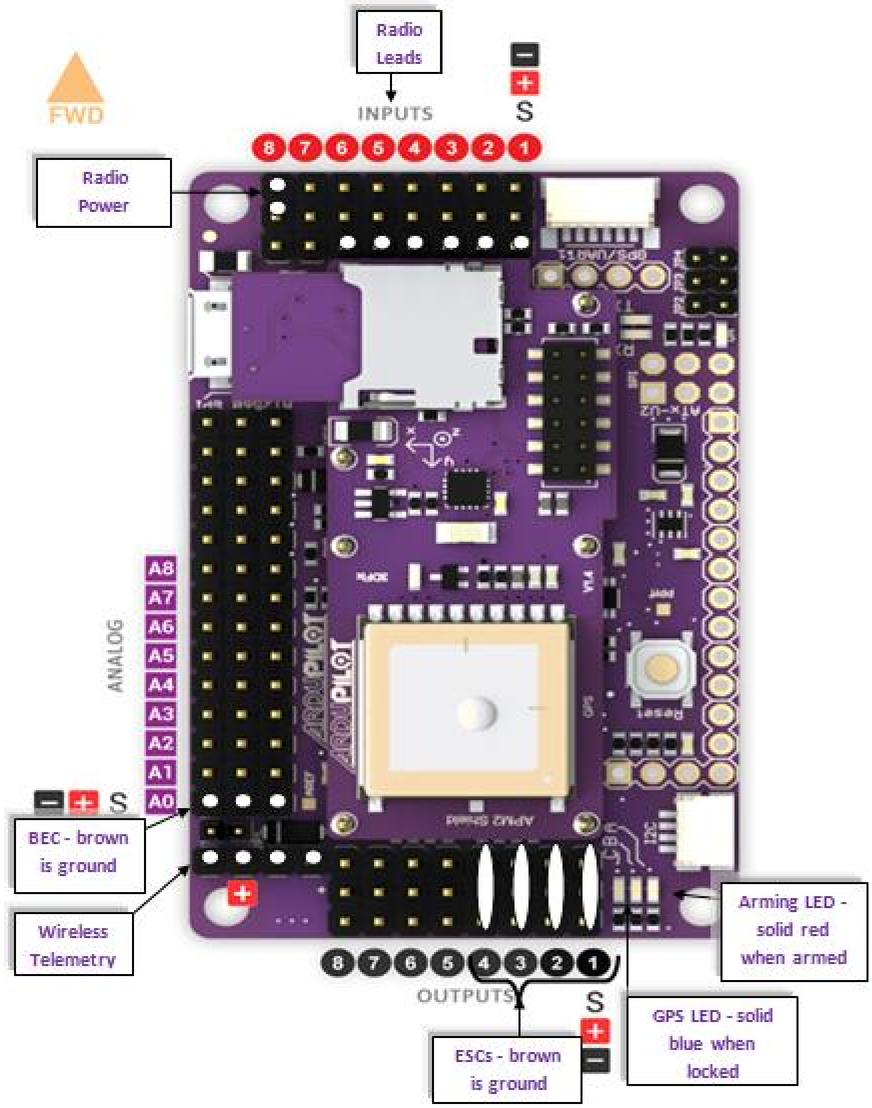
The APM2 features 8 Pulse Width Modulation (PWM) outputs and can be increased to 11, if you give up 3 of your 12 analogs and 8 PWM inputs. You can also bypass one of the pins with a solder jumper to insert your own PPM signal; still you can use the other PWM inputs left to control something else so you can have more than 8 inputs. The shipped quads only require 4 of the 8 PWM inputs.

The +5V servo power could be separated from the rest of the board, you could join both powers by insert a regular jumpers. This saves a lot of problem in some setups. It also features a protection diode to protect the board from reverse polarities. Reset pins are left exposed with ground so you can add an external reset switch if you wish.



The firmware is in the repository[9] and supports both APM1 and APM2. The Mission Planner will auto detect your board and load the appropriate firmware. The official 3DR APM2 board has a unique signature and the Mission Planner will look for that. All supplied APM2 boards should have firmware preloaded on them.

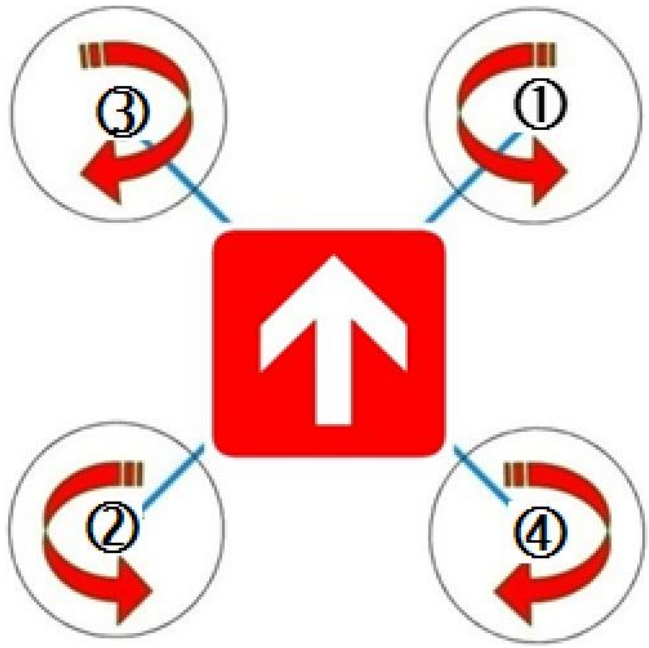
[9] http://code.google.com/p/arducopter/wiki/ArduCopter



The picture above shows how the APM2 is connected to the DJI ESCs, the Spektrum radio receiver, the 3DR wireless telemetry, and the BEC.

X-frame

All the quads are in an X configuration. The graphic below shows how the motors are numbered and the directions that they spin in.



If a motor is spinning in the wrong direction unplug any two of the black wires from the motor to the ESC and switch their positions. This will change the direction the motor spins. If the quad cannot takeoff and repeatedly flips over one motor, that motor is probably spinning in the wrong direction; its propeller cannot generate lift because it is spinning the wrong way.

PIDs

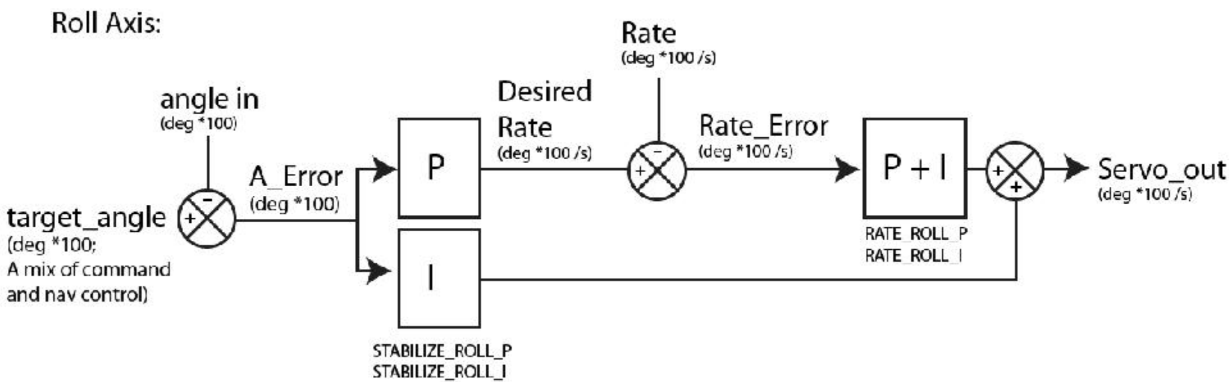
The APM2 controls the quad through the use of Proportional Integral Derivatives (PIDs). The following is quoted from the wikipedia.org page about PIDs:

"A Proportional Integral Derivative controller (PID controller) is a generic control loop feedback mechanism (controller) widely used in industrial control systems as a PID is the most commonly used feedback controller. A PID controller calculates an "error" value as the difference between a measured process variable and a desired setpoint. The controller attempts to minimize the error by adjusting the process control inputs.

In the absence of knowledge of the underlying process, a PID controller has historically been considered to be the best controller. By tuning the three parameters in the PID controller algorithm, the controller can provide control action designed for specific process requirements. The response of the controller can be described in terms of the responsiveness of the controller to an error, the degree to which the controller overshoots the setpoint and the degree of system oscillation. Note that the use of PID algorithms for control does not guarantee optimal control of the system or system stability.

Some applications may require using only one or two actions to provide the appropriate system control. This is achieved by setting the other parameters to zero. A PID controller will be called a PI, PD, P or I controller in the absence of the respective control actions. PI controllers are fairly common, since derivative actions are sensitive to measurement noise, whereas the absence of an integral term may prevent the system from reaching its target value due to the control action".

We have tuned the PID controller for the quad so you shouldn't have to do much work to obtain the desired flight characteristics. That being said, the quad will have to be tuned for different environmental factors; ours were tuned at 90m above sea level (Montgomery County Community College), different load distributions, and to optimize flight time. The following is adapted from the ArduCopter wiki explaining how the PIDs influence control of the quad:



The controller for the ArduCopter is a variation on the above idea. We take the angle error and generate a desired rate of rotation. This rate of rotation (in degrees) is compared to the current rate and multiplied times Rate P. The value is converted into Pulse Width Modulation (PWM) and sent to the motors. This gives us numerous benefits such as being able to specify the desired rate of travel, rotation or altitude change. It also gives us a second I term to control how must we compensate to achieve the desired rate.

Stabilize is used for all modes except "Acro Mode" (more on modes in Chapter 4). It is intended to be symmetrical, but pitch and roll have been broken out so you can fine tune a quad that is bearing a load in one axis. This might be a result of using two battery packs hanging on either side of the frame, or a very long thin pack. If one axis has a higher moment of inertia than the other axis, your ideal tuning parameters will not be symmetrical.

- **STABILIZE_P**: The rotation rate at which you want to correct any errors. The higher the number the faster the quad will attempt to achieve the desired attitude.
- **STABILIZE_I**: Used to account for Center of Gravity (CG) variations, weak motors or persistent external forces.
- **STABILIZE_IMAX**: Maximum amount the quad can compensate for these imbalanced forces.
- **RATE_P**: The most important value! This gain controls how much thrust you need to output to achieve the desired rate of rotation. Our high thrust/weight quad requires a lower value than a low thrust/weight quad would require. A too-high value will oscillate around 5-10Hz. You want this value to be slightly lower than the value that causes the oscillation. Aggressive tuning may result in exactly one overshoot and return, which is considered acceptable. Use channel 6 (CH6)

tuning to adjust in the air for best performance. For information on CH6 tuning can be found online in the ArduCopter wiki.

- **RATE_I**: Used to help a quad achieve a desired roll rate. Not used by default as this can be very difficult to tune properly and can be confusing.
- **RATE_IMAX**: The maximum amount of Rate_I that can build up. Having a zero iMax will make Rate_I completely ineffective, no matter how high the Rate_I is. Yaw is used to hold a particular yaw angle. If your quad wants to spin naturally, you won't be able to hold an exact heading. You will instead drift a few degrees until P gets significantly high to stop rotation.
- **STABILIZE_YAW_P**: The desired rate at which the quad will return to the target heading. If this is too high, it could cause an oscillation.
- **STABILIZE_YAW_I**: Acts like a trim to overcome poor quad balance. Defines time it takes to achieve max value. Higher = faster.
- **RATE_YAW_P**: Used to the amount of control authority the ArduCopter can use to achieve a zero yaw rate. If this is too low, you will never be able to stop a rotation. If this is too high, it will "yaw-oscillate. "
- **RATE_YAW_I**: Not used.

Loiter is used to control how much the quad will pitch towards the loiter target while trying to hold a position.

- **LOITER_P**: The rate at which the quad will move towards the target point. If this is not high enough, the quad will not be able to fight high winds and will drift. If it's too high, it will oscillate around the target.
- **LOITER_I**: This will help the quad fight winds while having a zero error. However use it with caution because it will also cause an oscillation if it's too high.
- **LOITER_IMAX**: The maximum possible buildup of LOITER_I.
- **NAV_WP**: Used to control the rate of speed of the quad towards the target.
- **NAV_WP_P**: we use our speed (defined by WP_SPEED_MAX) offset as the error. high numbers = more pitch to achieve the desired speed
- **NAV_WP_I**: Allows us to ramp up against the wind. Higher value ramps faster.
- **NAV_WP_IMAX**: Amount of Pitch we can add to overcome wind

Altitude Hold is used to hold a position using the relatively noisy Barometric pressure sensor.

- **ALTITUDE_HOLD_P**: Used to convert altitude error in centimeters to a desired climb rate in centimeter/second. Higher = faster climb rate.
- **ALTITUDE_HOLD_I**: Used to account for a quad having trouble holding altitude, usually due to a low voltage battery.
- **ALTITURE_HOLD_IMAX**: Amount of throttle we can adjust (units: 1000 = 100%)

Throttle Rate is used to dampen the quad and control the rate of climb.

- **THROTTLE_RATE_P**: amount of throttle output used to change the climb rate
- **THROTTLE_I**: compensates for error in achieving desired climb rate (zero by default as we use ALT_HOLD_I to do most of the work).
- **THROTTLE_IMAX**: Amount of THROTTLE_I we can add or remove to achieve desired climb rate.

The RATE_ROLL_P and RATE_PITCH_P terms go a long way to making your quad perform well, and are highly dependent on the thrust to weight ratio of your quad. More thrust = lower gains.

Chapter 4 – The APM Mission Planner

The best way to interact with the quad, aside from the RC controller, is through the APM Mission Planner. The Mission Planner is a Graphical User Interface (GUI) that allows you to plan missions, change parameters in the APM2 (including PIDs), configure the APM2 to your quad, load new firmware to the APM2, and view real time data on the quad via wireless telemetry.

One of the features of the Mission Planner is that it can run Python[10] scripts, so you can easily program your quad to do anything, from robotic acrobatics to just script-driven missions. We do not suggest this as getting the code correct is difficult and getting it wrong can end in an out-of-control quad. We suggest sticking to the Mission Planner's mission planning.

When opening the Mission Planner the screen should look like the screenshot below. If the Mission Planner is not loaded on your Acer tablet follow the instructions here:

https://code.google.com/p/ArduCopter/wiki/AC2Installation

The Mission Planner is organized according to tabs. The tabs in the screenshot above are in the red box. There are 7 tabs in all: Flight Data, Flight Planner, Configuration, Simulation, Firmware, Terminal, and Help.

- **Flight Data** has all the real time data from the APM2.
- **Flight Planner** is where you plan and write missions to the APM2.
- **Configuration** holds all the configurable items in the APM2.

[10] http://code.google.com/p/ardupilot-mega/wiki/Python

- **Simulation** has the ability to test out changes to the APM2 before flight. We have not used this simulator and cannot back up its ability to correctly simulate the quad.
- **Firmware** is where you load the most current firmware onto the APM2. To load ArduCopter 2.7 see instructions in the firmware section below.
- **Terminal** opens up a serial monitor that allows you to talk directly to the APM2, bypassing the GUI of the Mission Planner.
- **Help** will direct you to diydrones.com.

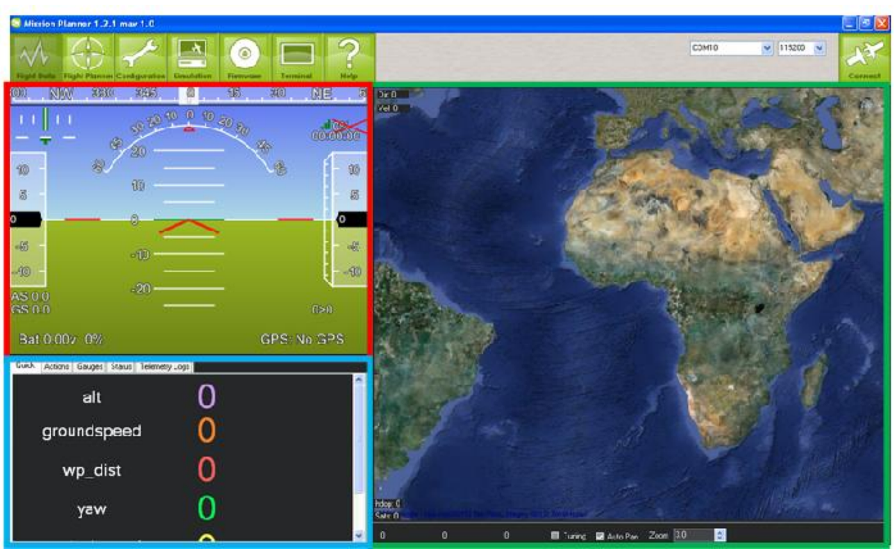
To connect to the APM2 use the section of the Mission Planner in the yellow box in the screenshot above. To connect via USB, plug the USB into the APM2; **do not have the battery connected at the same time**, and into your computer. Select the correct COM port from the drop down menu as an incorrect COM port will not allow you to connect, set the bit rate to the right of the COM port dropdown to “115200” and hit the “Connect” button. To connect wirelessly, plug the ground 3DR radio into your Acer tablet’s USB port and make sure the quad is powered by the battery as it will not wirelessly connect if you are plugged into the USB port. Select the correct COM port from the drop down menu, set the bit rate to “57600”, and hit the connect button.

If you are unable to connect to the quad double check that the COM port and bit rate are set correctly for the way you are trying to connect (USB or wireless). If they are correct and you still cannot connect make sure the bit rate of the Acer tablet’s COM port is set to the correct number (the same number used in the Mission Planner). To do this go into your device manager and find the COM port for your device; if you unplug the device the COM port will disappear. From here go into properties -> port settings and change the bit rate to the same number as used to connect in the Mission Planner (115200 or 57600). If you still cannot connect make sure the radios from 3DR are configured exactly the same. This is done in the configuration tab of the Mission Planner.

Flight Data

The Flight Data tab consists of 3 sections as shown below:

The red section is the Head’s-Up Display (HUD). This is shows the current data from the quad, as long as you are connected to it. The green section shows the current GPS position of the quad, as long as the APM2 has a GPS lock. The blue section has multiple functions which will be explained later.

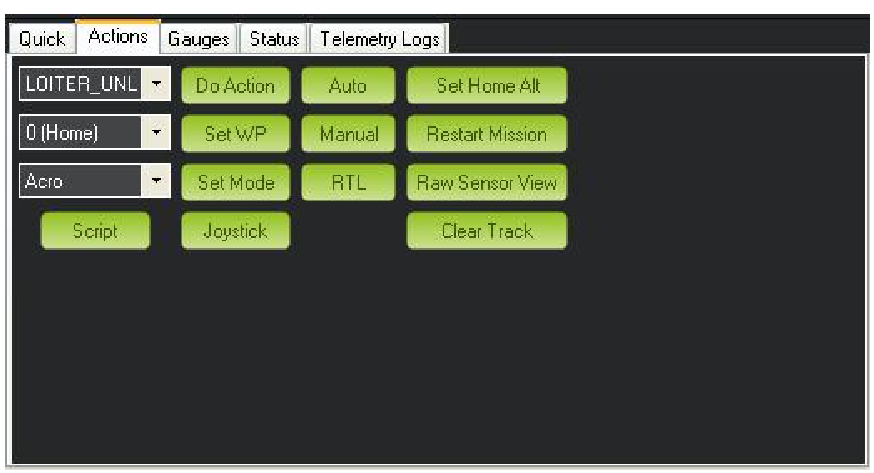


The red and green sections are detailed below:

Note: Groundspeed and Airspeed will always be the same because you cannot connect an airspeed sensor to the quad.

The blue section is split into 5 tabs as shown below. The first tab, “Quick” gives the user the ability to quickly see altitude, groundspeed, distance to next waypoint, yaw angle, vertical speed, and distance to MAV.

The second tab, “Actions” gives a list of actions you can tell the quad to do via the Mission Planner and the telemetry.



While it is possible to control the quad through this tab, we do not suggest it because it has not been extensively tested. The “Gauges” tab takes the barometric pressure sensor (VSI), speed, altitude and heading information presented in the HUD and presents it as four circular gauges. The “Status” tab shows the data coming from the quad. This data is split up into 120 different columns. “Status” gives you the best representation of data coming from the quad. Lastly is the “Telemetry Logs”. This tab gives you the ability to playback logs (text files generated by the Mission Planner every time it is turned on) of a mission using the Load Log button and the Play button. The Tlog, Kml and Graph buttons are not currently working.

Flight Planner

The Flight Planner tab allows users to plan an autonomous mission. It uses satellite imagery (whose source can be changed from the drop down menu on the middle right of the window) to allow for point and click mission planning. This tab also gives users the ability to save, load, read, and write missions, and a list of commands for planned missions. For more information on planning missions refer to the APM Mission Planner 1.2.1 Flight Document.

Configuration

The Configuration tab is where you are able to configure the aspects of the APM2. When not connected to the APM2, this tab will show only three selections to the left of the screen: 3DR Radio telemetry, Antenna Tracker, and Planner. The 3DR Radio and Antenna Tracker selections are used to change the settings on the 3DR Radios. The Planner selection allows users to customize the Mission Planner itself, including having the Mission Planner talk to you, as well as changing the units that numbers are represented in the HUD. When connected to the APM2 the Configuration tab allows user to change the parameters of the APM2.

When connected to the APM2, the Configuration tab… [To be completed after 10AUG2012].

Flight modes

The APM2 can fly in a variety of flight modes that can be mapped to the three position switch on the Spektrum radio transmitter. With the APM2 connected and the radio transmitter turned on, go into the Flight Modes section and you will see a list of 6 possible modes. Move the transmitter’s 3 position stick to see which modes (1 through 6) become selected; green means the mode is selected. Once you know which modes will be selected by which stick position you can change the modes.

The APM2 can fly in the following modes: Acro, Stabilize, Simple, Alt Hold, Position Hold, Loiter, Auto, RTL, Guided, Land, Circle, Toy, and OF_LOITER.

Acro, Stabilize, and Simple all give you total control of the quad.

Alt Hold, Position Hold, and Loiter only give you partial control of the quad. Increasing or decreasing the throttle stick will increase or decrease the altitude of the quad while moving the pitch and roll stick will change the position of the quad. Alt Hold only holds the quads current altitude, not its current position.

Auto, RTL, Guided, and Land are fully autonomous modes.

Circle, Toy, and **OF_LOITER** should not be used because they are not fully implemented and have not been tested.

More details about these flight modes:

Acro, Stabilize, and **Simple** modes all give the pilot total control of the quad, to varying degrees. Acro gives you the most control over the quad as the APM2 does the least amount of work to help you fly. In this mode the APM2 only uses the gyro sensor to control the craft. As such the quad will not self-stabilize. This mode is the hardest to fly in. In Stabilize mode the APM2 uses all the sensors attached to it to keep the quad as stable as possible. This is the mode you will primarily be flying in as it is the easiest total control mode to use. Simple mode is less of a flight mode and more of an addition to the selected mode. It allows the pilot to not worry about the quad yawing 180 degrees, which would invert the controls, because the APM2 translates the pilots stick inputs so that pitch up is always away from the pilot, down is towards, roll to the left will move the quad to the left of the pilot and roll to the right toward the pilot’s right. Simple mode should be selected along with Stabilize for the manual flight mode.

Alt Hold, Position Hold, and **Loiter** modes give the pilot partial control of the quad. Alt Hold mode will hold the current altitude of the quad. It does this, partially, by taking the current throttle stick input as the amount of throttle needed to keep the quad at the current altitude. Because of this, if you are descending or ascending when you switch into Alt Hold the quad will not hold altitude. The throttle stick would have been too high or too low so the mode would make the quad ascend or descend. With the quad holding altitude you can move it around using the sticks as you normally would, except for the throttle stick. The throttle stick no longer controls the throttle; it controls the height of the quad. Increasing the throttle will make the quad ascend and decreasing the throttle will make it descend. Note that there is a large central dead band around the throttle stick so you will have to move the throttle past this band to have the quad change altitude. Position Hold and Loiter mode are the same as Alt Hold except they will also try to hold the quad’s GPS position. Note that you can move the held position by moving the roll and pitch sticks and that you cannot arm the motors in these modes.

Auto, RTL, Land, and **Guided** modes are fully autonomous modes. Auto mode will make the quad perform its preprogrammed mission. RTL will cause the quad to return to its set home position at the altitude the quad was at when RTL was enabled. Land will cause the quad to land at its current position. Guided mode should allow you to tell the quad where to go by right clicking on the Flight Data tab’s satellite map and clicking “Fly Here”. This mode has not been tested; use it at your own risk. For all of the autonomous modes do not drop the throttle stick unless the quad has touched the ground in Land mode as this could cause the quad to drop out of the sky. Note that you cannot arm the motors in these modes.

Circle, Toy, and **OF_LOITER** should not be used because they are not fully implemented and have not been tested.

RTL, Loiter, Circle, and **Land** are also commands for autonomous missions. The APM Mission Planner 1.2.1 Flight Manual gives more details on these commands.

Simulation

The Simulation tab is used to run simulations to check new parts of the ArduPilot code or to test PID configurations. We have not dealt with this simulator and thus cannot guarantee that it will accurately reflect reality. For more information on the simulator, go to diydrones.com.

Firmware

The Firmware tab is where you load the most current version of the ArduCopter firmware onto your APM2. We do not suggest using this method to load firmware onto your APM2 because we have not tested the firmware version you could be loading, we have only tested version 2.7. To load the supported firmware you must flash the APM2 with the ArduCopter.pde file in the ArduCopter 2.7 folder. To do this plug the APM2 into your computer using the USB cord. Then open the version of the Arduino IDE installed on your computer, it should say relax patch in its name, and load the ArduCopter.pde file. Make sure the COM port in the Arduino IDE is set to the COM port of the APM2 and the bit rate is set to 115200. From there compile the code in the IDE. There will be many red/orange error messages but the compile should not stop. With this done, connect to the Mission Planner as you normally would. If you are unable to connect there is a chance the firmware was not compiled correctly.

Terminal

The Terminal tab is where you can talk to the APM2 without the interference of the GUI of the Mission Planner. Though it should not be necessary, if the Mission Planner is not working correctly; for example you cannot configure something. You will have to use the Command Line Interface (CLI) mode of the APM2, which is implemented through the Terminal tab.

With the APM2 connected through the USB and the correct COM port and bit rate selected -but not yet connected- go into the Terminal tab. After a few moments it should ask you to hit enter three times to enter CLI mode. Hit Enter three times and it should give you something like this: [ArduCopterv2.7]. This means you are in CLI mode.

A complete list of CLI commands is found here:

https://code.google.com/p/arducopter/wiki/AC2CLI?wl=en

Help

The Help tab gives you basic information on the tabs and directs you to internet resources. It also allows you to check for updates to the Mission Planner. We do not suggest updating the Mission Planner as we have not tested the Mission Planner you could be loading, we have only tested version 1.2.1.

Chapter 5 – Troubleshooting

In this Chapter, you will find a troubleshooting guide adapted from diydrones.com.

I can't get the motors to run and/or can't arm the ESCs

This can be due to a number of mistakes:

1. First, if you're using APM 2, press the reset button after you first plug in your LiPo to ensure a proper boot. This will be fixed in the next version of the code.
2. Are you in Acro, Stabilize or Simple mode? As a safety measure, the motors won't arm in other modes.
3. Are you sure you're arming them right? When the motors are armed, the arm LED is yellow on APM2, on the IMU it will turn solid after the IMU calibration flashing inabout 5 seconds. Review the arming guide[11] to make sure you're doing it right.
4. Is your "Yaw" or "Throttle" channel reversed on your Spektrum transmitter? Try moving the Yaw stick to the far left, instead of the right. If that works the green LED turns solid after the IMU calibration, your Yaw is reversed. If not, try the same with the Throttle channel: push it to max and try full right Yaw from there. If that works, your Throttle channel is reversed.
5. It's possible you've changed your throttle trim settings since you did your initial radio calibration. Trim the throttle down a few notches and try arming again.
6. Are you flying a Mode 1 transmitter? If so, the throttle is not on the Yaw stick. Make sure it's fully down when you're pushing the Yaw stick to the side.
7. Is your RC throttle channel reversed? Connect one of the ESCs directly to your RC receiver Channel 3 (CH3) and plug it into a LiPo -take the propellers off- and try it manually to be sure.
8. Redo your radio calibration in the Mission Planner or CLI. Make sure that the throttle CH3 is being read properly and the values are around 1100 (low) and 1900 (high), more or less.
9. If you've checked the previous things, it may be that you need to recalibrate your ESCs. Do it the manual way.

When I power my APM2 with a battery, the blue LED comes on but the other LEDs don't flash as part of the calibration process

This is a known issue that we're resolving. In the meantime, just press the reset button once and it should boot normally.

My APM 2 is locking up

The cause is probably that the dataflash card not initializing correctly. To verify this is the problem, take it out and try to power your APM2 again. You should be able to go through the tests in the CLI

[11] https://code.google.com/p/arducopter/wiki/AC2_Flying

successfully. If this was indeed the case, there is an easy fix. Put the dataflash back in, and power the board up while looking at the terminal screen. It will reformat the card -this will take a while, be patient. Leave it plugged in for a good 5-10 min. After that you should see a “ready to fly” message. You only have to do this once.

ArduCopter tilts/flips over or wobbles crazily when I try to take off.

This almost always means that you've configured it or set it up wrong. Check the following:

1. Is the APM board facing forward and in the direction of forward motion (towards the forward arm in + mode and between the arms in x mode)?

2. Have you set your quad orientation right in the setup process: + or x-frame?

3. Have you run the Motors command in the CLI setup to ensure that all the motors are hooked up right, turning the right way and the pusher/puller props are on the right motors and oriented the right way? You can do this using a piece of paper held against the motor to see which way it is turning.

4. Check that your propellers can't turn on their motor shafts, as if they weren't tightened down enough. If one can, despite its propeller being tightened, that may mean that its motor shaft has been pushed down into the motor too far for the prop mount to get a grip. Take off the propeller and use a small Allen wrench to loosen the little set screw that holds the shaft onto the motor. Then with pliers pull the shaft out until it's the same height as those of the other motors. Then tighten the set screw, and put the prop back and tighten it up again.

5. Was the quad stable and flat for the whole calibration routine at startup? Was it stable and flat during the setup process? If need be, do it again.

6. Did you calibrate your ESCs? They should all start together when you advance the throttle. If need be, do it again manually.

7. Did you setup and test your radio? Go into Configuration tab and transmitter section to see. Pitch, Roll, Throttle and Yaw should all be 0 or near zero (60 or less). Move the pitch and roll stick to the lower right, you should see approximately 4500, 4500 in CH1 and CH2.

8. Have you checked all your sensors in the Mission Planner or CLI Test/IMU menu to make sure the hardware is working right?

My radio setup isn't reading the RC channels right or hangs

Check the following:

1. Are you sure you're powering the RC pins with an external power source? The USB cable will NOT power the RC receiver. You need to have a LiPo plugged into your power distribution board.

2. Is your battery fully charged? The PPM encoder is more susceptible than the rest of the APM to an under voltage situation. If you're getting the "slow blink” on your blue PPM encoder LED when the radio is connected and powered, try switching in a freshlycharged LiPo.

3. Double-check the soldering! A tiny short on the APM board could disable all the channels.

4. We had reports of people having trouble with Spektrum radios, which have a funky binding process. This is what works: To bind the receiver and telemetry see the Spektrum manual.
5. Check your cables! The signal wire should be at the top, and ground at the bottom.
6. Make sure your transmitter is on!
7. Finally, if all else fails, try to narrow down the problem. First, check that the receiver is working right by plugging a servo into the output you're testing and making sure that it's outputting a good signal. Then plug one known-good RC channel into the APM input channels, from CH1 to CH7, one at a time and see if any are read. If some channels are read and some are not, you may have a soldering error or may have fried one or more inputs.

My quad isn't as stable as I'd like or doesn't Loiter/Hold Altitude/Navigate reliably

Every quad is different, and even could have differences in motor performance or balance depending on how it was put it together. So we've made it relatively easy to tweak many of the performance settings to turn your quad perfectly. That said, tuning is something of an art, so it helps to read a bit more before you start tweaking.

My quad flies okay, but it tends to tilt one way or another

Make sure the Center of Gravity (CG) of the quad is dead center. Then run the level command on a flat surface and hold/disarm for ~15 seconds to invoke. You can also fly in auto-trim mode in a windless (important!) environment -wind will cause the changes you make to work against you when the quad rotates 180°. Hold arm for ~20 seconds to fly in auto-trim mode for about 45 seconds. It will exit auto-trim automatically and fly normally. Don't worry about counting out the time, just wait for the flashing lights to enter either mode.

You can use your radio pitch and roll trims, but remember to re-center them when you set up your radio with the Configuration tools. I'm not a fan of trimming the radio for quads, but never, never trim yaw. Quads are also susceptible to drafts. They will need constant correction unless you install an optical flow sensor.

The quad always wants to yaw to the right or left when I take off

This is usually due to an airframe that it out of tune, either with one motor tilted slightly or the weight balance not centered. Typically, the bad behavior is that the quad will always yaw by a certain amount, typically around 30-45 degrees on takeoff, but will stay in that direction. This is because the mechanical asymmetry of an out-of-tune quad is forcing the yaw, and the limit is just how long it takes for the P term in the AC2's PID equations to grow and stop it.

Although you can adjust those terms in the Mission Planner's PID configuration screen, the best solution is to solve it at the source. Eyeball each motor and see if it or the arm it is one is slightly tilted, and bend it back to vertical if so. Also ensure that the battery is centered on the quad and the center of gravity is as close to the center of the APM board as possible. It's also a great idea to redo the ESC calibration routine just in case.

Also make sure that you've got your forward-rotating and counter-rotating props on the right motors.

The quad always wants to yaw when I pitch or roll

Your compass may need calibrating. AC2 calibrates while flying to take the magnetic fields of your motors into account. You must disarm the motors at the end of your flight to save your calibration to EEPROM. It usually takes 1-2 flights before the calibration makes a difference.

You must use a compass in conjunction with a GPS. If you disable your compass, please disconnect your GPS or you will have random Yaw hold issue.

Sometimes my quad spins around 180 degrees all by itself!

This is most commonly due to people not reading the manual and using a GPS without also using and enabling a magnetometer. The GPS module generates a “directional vector” based on motion over ground. Unlike an airplane, a quad can fly backwards, so the GPS will think it has reversed direction even through it's still pointed in the original direction.

That's why we require that you use a magnetometer when you're using GPS. Even if the GPS is confused about what direction the copter is pointed, the magnetometer will not be. When the magnetometer is installed and enabled, the ArduCopter code will follow it for orientation information, not the GPS.

My quad flies well, but then dips a motor arm in a fidgety manner while hovering

The most likely cause of this are your bullet connectors. There are some other things that can cause this problem. Your motor may be going bad: either the bearings are going or your shaft is horribly bent. A motor with bad bearings takes more power to spin. In that situation, the ESC could be cutting out to protect itself. Or it may be flying slow enough to stall the motor. Attach a current sensor between your battery and the motor and test the difference between a good motor and the bad one. If you are seeing higher draw in the bad one, replace or fix it.

In the “raw sensor view” of the Mission Planner, the Z accelerometer reading is always 1000, not 0, when it's on the bench

That's normal. The Z accelerometer is showing the force of gravity which is an acceleration, as you may recall from high school physics. If you turn the quad upside down, you'll see that it reverses.

I can't connect with the Mission Planner over USB

If you are unable to load firmware or otherwise connect with AC2 via USB, please double-check that you have followed all the procedures here: https://code.google.com/p/ardupilot-mega/wiki/DownloadCode. If you are using Windows 7, the FTDI drivers should have been automatically loaded, but if not you must download them yourself as described in those instructions. Double check that you've selected the right COM port and 115200 as the bit rate in the Mission Planner.

If you're still having trouble, check the troubleshooting guide[12] and finally, try another PC and/or another USB cable.

[12] http://diydrones.com/profiles/blogs/arduino-debugging-tips

My quad's ESCs keep beeping

1. Have you set up your RC input in the Mission Planner setup process?
2. Check that your radio is talking properly to its receiver and the small blue LED in the lower red APM2 board is blinking fast.
3. Check that your radio channels are in right order, cabling and/or Mode-1, Mode-2.
4. You can use PWM test function from the CLI to see that your channels are correct. Especially check CH3, it should be around 1100 when your throttle stick is down and around 1900 when up.

I'm not getting GPS lock at all

First, note that you will probably only get GPS lock outside. Iif you're very near a window or are lucky enough to have strong GPS signal and a radio-transparent roof, you may be able to get lock inside. For a cold start (power on), it make take three minutes or more. If you're just hitting the reset button after lock has been achieved, it should take less than a minute.

If you can't get GPS lock outside after many minutes, the blue LED on the MediaTek module keeps blinking, you may have a defective module. If the module shows lock, but the red APM2 GPS LED is still blinking, you either have a bad cable or an older MediaTek module that needs to have its firmware updated[13].

I want to load the code in Arduino, but I'm having trouble

First, ensure you've downloaded the latest AC2 code[14], unzipped it to a folder on your desktop, and that you are using the latest Arduino, at least 022.

If you're getting compile errors that means that you haven't told Arduino where your libraries are properly. You can use the similar instructions for the APM2[15] to show you how to do that.

If you're getting "avrdude" sync error message in the Arduino status bar, go through all of the tips[16].

My logs aren't working

First, erase the logs in the CLI "logs" menu

By default, the most common data fields are enabled for logging.

In the CLI "logs" menu you can turn each field on or off. Here are the defaults:

LOG_ATTITUDE_FAST	DISABLED
LOG_ATTITUDE_MED	ENABLED
LOG_GPS	ENABLED
LOG_PM	ENABLED
LOG_CTUN	DISABLED

[13] https://code.google.com/p/arducopter/wiki/GPSFirmware
[14] http://code.google.com/p/arducopter/downloads/list
[15] code.google.com/p/ardupilot-mega/wiki/Code
[16] http://diydrones.com/profiles/blogs/arduino-debugging-tips

LOG_NTUN	DISABLED
LOG_MODE	ENABLED
LOG_RAW	DISABLED
LOG_CMD	ENABLED
LOG_CURRENT	ENABLED

If the Mission Planner or CLI doesn't show logs available to be read, go to the CLI logs menu and type “dump 0”. It should dump data for a few minutes, and then you can select all and copy and paste it into a text editor. If you save it to your hard drive with the extension “.log” you can read it and analyze it with the Mission Planner in the Terminal's “Log Browse” button.

If the CLI or Mission Planner is still unable to read any data at all, you may have a soldering problem.

My quad just won't rise/lift off

If you have a problems on getting your quad up from the ground, check that you propellers are mounted correct way.

My quad feels sloppy on roll or pitch axis

Both of roll and pitch axis should give to you exact or close to similar response when doing hand tests. If one of the axes does not respond as expected, check your PID settings and you can try to upload firmware again.

Firmware upload might be successful but due internal timing issues some of the code is not written properly and this can cause unstabilities on your quad.

My quad oscillates slowly (larger movements) when stabilized

Lower your gain in STABILIZE_ROLL_P.

My quad oscillates quickly (smaller movements) when stabilized

Lower your gain in RATE_ROLL_P.

My quad gets wobbly when descending quickly

Raise your gain in RATE_ROLL_P. You can tune most of this out, but it’s impossible to descend without some wobbles.

My quad is too sluggish

Raise your STABILIZE_ROLL_P gain. These gains make ask the quad to respond faster to angle errors.

My quad yaws right or left 15° when I take off

Your motors are not straight or your ESCs are not calibrated. Twist the motors until they are straight. Run the ESC calibration routine.

After aggressive flying my quad leans to one side 10-30°

Do whatever you can to remove vibrations and isolate APM. If you land for a few seconds, then continue flying, it will give APM a chance to correct without the vibration.

My quad won’t stay perfectly still in the air

Run the level command on a flat surface (hold disarm for 15 seconds to invoke). You can also fly in auto-trim mode in a windless (important!) environment. Any wind will cause the changes you make to work against you when the quad rotates 180°. You can use your radio pitch and roll trims, but remember to re-center them when you set up your radio with the configuration tab in the Mission Planner. Never Trim Yaw, your quad may start spinning on its own. Quads are also susceptible to drafts. They will need constant corrections unless you install an optical flow sensor (not included on shipped quads).

My quad flies well, but then dips a motor arm in a fidgety manner while hovering

Your motor may be going bad or the bullets connectors between the motor and ESC may be at fault. Vibrations from a bent shaft or unbalanced propellers can make bullet connections fail momentarily, stopping the motor. A motor with bad bearings takes more power to spin which may cause the ESC to cut out to protect itself. Attach a current sensor between your battery and the motor and test the difference between a good motor and the bad one. If you are seeing higher draw in the bad one, replace it or fix it.

In loiter, my quad constantly overshoots

Try and increase your NAV_P term. You can also tune down your I terms because in some cases, the NAV_I or LOITER_I can causes overshoot. Making the I terms zero when in still wind is the best way to tune NAV_P.

My altitude hold above 3m is only about 1-2m accurate

That’s actually the best you can achieve. The barometric sensor is sensitive to light and wind.

My quad increasingly swings up and down in “Auto Mode” and eventually hits the ground

Your THROTTLE_P is too high or low. You don’t need a lot of P to do altitude hold. Think of how much you move the throttle to hold altitude perfectly. That’s what you need P to do. I will ramp up as your battery goes lower to make up the difference.

My quad loiters by rotating in a clockwise (CW) or counterclockwise (CCW) circle: Adjust your compass declination until it stops circling.

THIS PAGE BLANK

Chapter 6 – Glossary

2.4GHz: The frequency used by digital (spread spectrum) radio communications in our applications, including 2.4GHz RC, Bluetooth and some video transmission equipment. This is a different band than the older 72MHz band that is used for analog RC communications. To avoid radio frequency conflict is it often a good idea to use 72MHz radio equipment when you are using 2.4GHz onboard video transmitters, or use 900MHz video when using 2.4GHz RC equipment. Our RC equipment transmit at 2.4GHz.

AGL: Altitude above ground level.

AHRS: Altitude Heading Reference System. An IMU (see below) plus the code to interpret the output from its sensors to establish a plane's XYZ and heading orientation.

APM: See ArduPilot Mega.

AMA: Academy of Model Aeronautics. The main US model aircraft association. Generally hostile to amateur UAVs, which are banned on AMA fields. But each AMA chapter and field may have slightly different policies, and it's possible to test airframes and some technology on AMA fields without violating the association's rules.

Arduino: An open source embedded processor project. Includes a hardware standard currently based on the Atmel Atmega168 microprocessor and necessary supporting hardware, and a software programming environment based on the C-like Processing language. See the official Arduino web site[17].

ArduCopter: Rotary-wing autopilot software for the ArduPilot Mega electronics.

ArduPlane: Fixed-wing autopilot software for the ArduPilot Mega electronics.

ArduRover: Ground and water autopilot software for the ArduPilot Mega electronics.

BEC: Battery Elimination Circuit. A voltage regulator found in ESCs (see below) and as a stand-alone product. Designed to provide constant 5V voltage for RC equipment, autopilots and other onboard electronics. Our BEC is standalone.

Bootloader: Special code stored in non-volatile memory in a microprocessor that can interface with a PC to download a user's program.

COA: Certificate of Authorization. A FAA approval for a UAV flight. See the FAA web site[18] for more details and the DIY Drones Regulatory FAQ web site[19]

[17] http://www.arduino.cc/en/
[18] http://www.faa.gov/about/office_org/headquarters_offices/ato/service_units/systemops/aaim/organizations/uas/coa/
[19] http://www.diydrones.com/profiles/blogs/regulatory-faq

DCM: Direction Cosine Matrix. An algorithm that is a less processing intensive equivalent of the Kalman Filter. See the DIY Drones web site[20] for more.

Eagle file: The schematic and PCB design files (and related files) that tell PCB fabs how to create the boards generated by the free Cadsoft Eagle[21] program. This is the most common standard used in the open source hardware world, although, ironically, it's not open source software itself. Needless to say, this is not optimal, and the Eagle software is clumsy and hard to learn. One hopes that an open source alternative will someday emerge.

ESC: Electronic Speed Control. Device to control the motor in an electric aircraft and serves as the connection between the main battery and the RC receiver. Usually includes a BEC, or Battery Elimination Circuit, which provides power for the RC system and other onboard electronics, such as an autopilot. The DJI ESCs do not include a BEC.

FPV: First-person view. A technique that uses an onboard video camera and wireless connection to the ground allows a pilot on the ground with video goggles to fly with a cockpit view.

FTDI: FTDI stands for Future Technology Devices International, which is the name of the company that makes the chips. A standard to convert USB to serial communications. Available as a chip for boards that have a USB connector, or in a cable[22] connected to breakout pins.

GCS: Ground Control Station. Software running on a computer on the ground that receives telemetry information from an airborne UAV and displays its progress and status, often including video and other sensor data. Can also be used to transmit in-flight commands to the UAV, although we do not suggest it.

HIMD: HIMD stands for Hard Iron Magnetic Distortion. Distortions that arise from magnets or magnetized metals on the airframe that affect the compass. These distortions remain in the same location relative to the compass for all heading orientations. These distortions can be mostly compensated for by adding constant offsets to the values returned from the 3D magnetometer.

HIL: HIL stands for Hardware-in-the-Loop simulation**.** Doing a simulation where software running on another computer generates data that simulates the data that would be coming from an autopilot's sensors. The autopilot is running and doesn't "know" that the data is simulated, so it responds just as it would to real sensor data. Hardware-in-the-loop uses the physical autopilot hardware connected to a simulator, as opposed to simulating the autopilot in software, too. In an "open loop" simulation, the software simulator feeds data to the hardware autopilot; in a "closed loop" simulation, the hardware feeds data back to the software simulator, too.

I2C: A serial bus that allows multiple low speed peripherals, such as sensors, to be connected to a microprocessor. See this wiki[23] for more.

IDE: An Integrated Development Environment, such as the Arduino editor/downloader/serial monitor software[24] and often includes a debugger.

[20] http://diydrones.com/forum/topics/robust-estimator-of-the
[21] http://www.cadsoftusa.com/
[22] http://store.diydrones.com/FTDI_Cable_3_3V_p/ttl-232r-3v3.htm
[23] http://en.wikipedia.org/wiki/I%C2%B2C

IMU: An Inertial Measurement Unit such as the ArduPilot OilPan[25] of the daughterboard on the APM2. Usually has at least three accelerometers measuring the gravity vector in the x, y and z dimensions and two gyros measuring rotation around the tilt and pitch axis. Neither are sufficient by themselves, since accelerometers are thrown off by movement, they are “noisy” over short periods of time, while gyros drift over time. The data from both types of sensors must be combined in software to determine true aircraft attitude and movement to create an AHRS (see above). One technique for doing this is the Kalman Filter (see below).

Inner loop/Outer loop: Usually used to refer to the stabilization and navigation functions of an autopilot. The stabilization function must run in real-time and as often as 100 times a second (the “inner loop”), while the navigation function can run as infrequently as once per second and can tolerate delays and interruptions (the “outer loop”).

INS: Inertial Navigation System. A way to calculate position based on an initial GPS reading followed by readings from motion and speed sensors using dead reckoning. Useful when GPS is not available or has temporarily lost its signal.

ICSP: In Circuit Serial Programmer**.** A way to load code to a microprocessor, usually seen as a six-pin (two rows of three) connector on a PCB. To use this, you need a programmer that uses the Serial Peripheral Interface (SPI) standard.

Kalman Filter: A relatively complicated algorithm that, in our applications, is primarily used to combine accelerometer and gyro data to provide an accurate description of aircraft attitude and movement in real time. See this web site[26] for more.

LOS: Line of Sight. See VLOS below.

LiPo: Lithium Polymer battery. This battery chemistry offers more power and lighter weight than NiMh and NiCad batteries.

MAVLink. The Micro Air Vehicle communications protocol used by the ArduCopter and ArduPlane lines of autopilot. See this web site[27] for more info on MAVLink.

NMEA: National Marine Electronics Association standard for GPS information. When we refer to “NMEA sentences,” we’re talking about ASCII strings from a GPS module that look like this: $GPGGA,123519,4807.038,N,01131.000,E,1,08,0.9,545.4,M,46.9,M,,47**.**

Oilpan: ArduPilot Mega IMU Shield[28]. An Arduino style shield meant to be paired with the APM1. Contains the majority of the sensors, gyros, accelerometers, barometer that allow the ArduPilotMega to function as an autopilot.

---

[24] http://arduino.cc/en/Main/Software
[25] http://store.diydrones.com/ArduPilotMega_IMU_Shield_OilPan_Rev_H_V1_0_p/br-0012-01.htm
[26] http://tom.pycke.be/mav/71/kalman-filtering-of-imu-data
[27] http://qgroundcontrol.org/mavlink/start
[28] http://store.diydrones.com/ArduPilotMega_IMU_Shield_OilPan_Rev_H_V1_0_p/br-0012-01.htm

OSD: On-Screen Display. A way to integrate data, often telemetry information, into the real-time video stream the aircraft is sending to the ground.

PCB: Printed circuit board. In our use, a specialized board designed and “fabbed” for a dedicated purpose, as opposed to a breadboard or prototype board, which can be used and reused for many projects.

PDB: Power Distribution Board. A board used in multicopters to distribute power to multiple ESCs. Our supplied quads use their lower baseplate as a PDB.

PIC: Pilot in Command. Refers to a FAA requirement that UAVs stay under a pilot’s direct control if they are flying under the recreational exemption to COA approval. See Line of Sight above. (Not to be confused with the PIC processor series by Microchip).

PID: Proportional Integral Derivative control method. A machine control algorithm that allows for more accurate sensor-motion control loops and less overcontrol. See this web site[29] for more.

RTL: Return to Launch. Fly back to the “home” location where the aircraft took off.

SiRF III: The standard used by most modern GPS modules including !SiRF III binary mode, which is an alternative to the ASCII-based NMEA standard described above.

Sketch: The program files, drivers and other code generated by the Arduino IDE for a single project.

SVN: Short for the Subversion Version-Control repository used by the DIY Drones and other teams for source code.

Thermopile: An infrared detector. Often used in pairs in UAVs to measure tilt and pitch by looking at differences in the infrared signature of the horizon fore and aft and on both sides. This is based on the fact that there is always an infrared gradient between earth and sky, and that you can keep a plane flying level by ensuring that the readings are the same from both sensors in each pair, each looking in opposite directions.

UAV: Unmanned Aerial Vehicle. In the military, these are increasingly called Unmanned Aerial Systems (UAS), to reflect that the aircraft is just part of a complex system in the air and on the ground. Ground-based autonomous robots are called Unmanned Ground Vehicles (UGVs) and robot submersibles are called Autonomous Underwater Vehicles (AUVs). Robot boats are called Unmanned Surface Vehicles (USVs).

VLOS: Visual Line of Sight. The pilot’s ability to see an aircraft from the ground well enough to control it, without the use of artificial visual aids, aside from glasses. Required by FAA regulations.

WAAS: Wide Area Augmentation System. A system of satellites and ground stations that provide GPS signal corrections, giving up to five times better position accuracy than uncorrected GPS. See this web site[30] for more.

---

[29] http://en.wikipedia.org/wiki/PID_controller

Xbee: The commercial name of the recommend ZigBee-compatible radio modems commonly used by amateur UAVs.

ZigBee: A wireless communications standard, which has longer range than Bluetooth but lower power consumption than WiFi.

[30] http://en.wikipedia.org/wiki/Wide_Area_Augmentation_System

THIS PAGE BLANK

Appendix 1 – Lithium Polymer Batteries

The quad comes with 5000mAh (milliamp-hour) 3-cell LiPo (Lithium Polymer) batteries. The "Complete Guide to Lithium Polymer Batteries and LiPo Failure Reports" below is adapted from a post by Jim McPherson of RCGroups.com.

Lithium batteries are the preferred power sources for most electric modelers today. They offer high discharge rates and a high energy storage/weight ratio. However, using them properly and charging them correctly is no trivial task. There are many things to consider before using lithium cells for e-flight. But none is more important than safety.

Charging and safety

Until you are willing to follow all safety precautions, DO NOT use lithium batteries. Lithium cells must be charged very differently than NiCad or NiMH. They require a special charger specifically designed to charge lithium cells. In general any charger that can charge lithium ion can charge lithium polymer, assuming that the cell count is correct. You must never charge lithium cells with a NiCad (Nickel Cadmium) or NiMH (Nickel-Metal Hydride) only battery charger. Charging lithium batteries is the most hazardous part of using them and extreme care must be taken when charging them.

It is important to set your charger to the correct voltage or cell count when charging. Failure to do this can cause the battery to spew violent flames. There have been many fires directly caused by lithium batteries. Please be responsible when charging lithium batteries.

1. **Use only a charger approved for lithium batteries.** The charger may be designed for Li-Ion or Li-Poly. Both batteries are charged in exactly the same way.

2. **Make certain that the correct cell count is set on your charger.** Watch the charger very closely for the first few minutes to ensure that the correct cell count continues to be displayed. If you don't know how to do that, read the manual provided with the charger.

3. **NEVER charge the batteries unattended.** This is the number one reason for houses and cars being burned to a crisp by lithium fires.

4. **Use a safe surface to charge your batteries** on so that if they burst into flame no damage will occur. Vented fire safes, Pyrex dishes with sand in the bottom, fireplaces, and plant pots are all good options.

5. **Do not puncture the cell, ever.** If a cell balloons quickly place it in a fire safe place, especially if you were charging it when it ballooned. After you have let the cell sit in the fire safe place for at least 2 hours, discharge the cell/pack slowly. This can be done by wiring a flashlight bulb of appropriate voltage (higher is voltage is OK, lower voltage is no) up to your batteries connector type and attaching the bulb to the battery. Wait until the light is completely off, than recycle the battery.

6. **If you crash, your lithium cells they may be damaged.** If you crash in any way carefully remove the battery pack from the aircraft and watch it carefully for at least the next 20 minutes.

7. **Charge your batteries in an open ventilated area.** If a battery does rupture or explode, then hazardous fumes and material will spew from the battery. Charging the batteries in an open ventilated area reduces the risk of these fumes and material harming others.

Lithium what?

LiPo batteries are used in many electronic devices. Most, if not all, LiPo batteries are not designed for RC use, we use them in different applications than they were designed for. LiPo batteries are similar to Lithium Ion batteries in that they each have a nominal voltage of 3.6 volts, but dissimilar in that they do not have a hard metal casing but rather a flexible material encloses the chemicals inside. The “normal” LiPo batteries are thin rectangle shapes with two tabs on the top one positive one negative. The reason we use them is that they are significantly lighter than comparable NiCad or NiMH batteries, which makes our quads fly longer and better.

Voltage and Cell Count

For a 5000mAh 3-cell LiPo battery, it is fully charged at 12.6V; the cut off is, functionally, at about 9.6V. When the battery reaches 9.6V do not fly with this battery until it has been charged. Exceeding these limits can harm the battery. We, currently, do not have a way to measure voltage while the quad is in flight

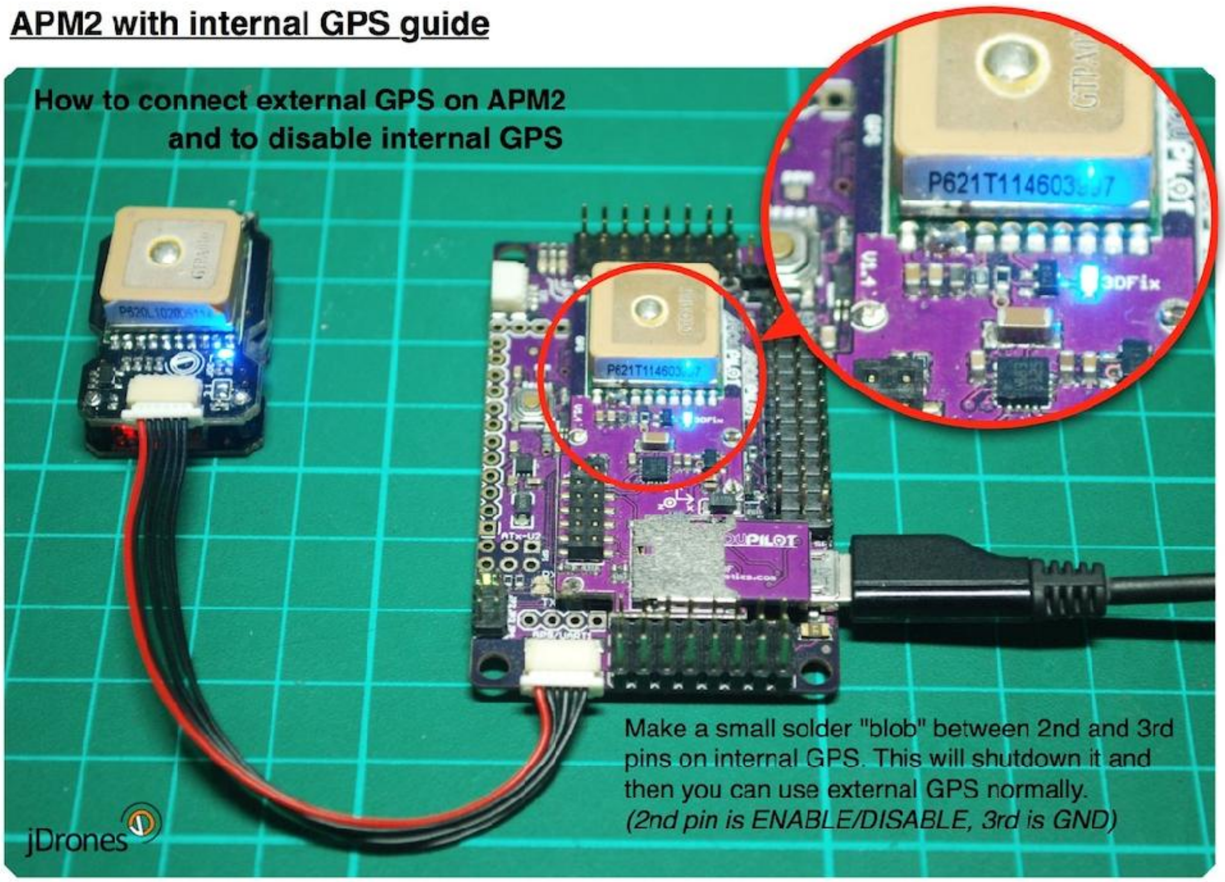
Dealing with temperature

LiPo batteries like heat, but not too much. In the winter time, try to keep your batteries from the cold as much as possible. Leave them in the car while you are flying, keep them in your cargo pants. At the same time don't let them heat up too much. Try to keep your batteries from reaching 70C after use. This will prolong the life of the cells. A good way to measure temperature is a handheld infrared (IR) meter.

Appendix 2 – Disabling the built-in GPS on APM2

If you have an APM 2 with the built-in Mediatek GPS and would like to use an external and/or a different GPS module, you can do so with some simple modifications.

Just solder a blob over these two pins to short them out.



Testing

1. With no external GPS, connect to APM2 via the Mission Planner. The HUD should report “No GPS”. That's good --it means it can't detect a GPS module. Now disconnect and remove the power from your board/remove the USB cable.
2. Connect your external GPS to the GPS port.
3. Connect the USB cable and connect via the Mission Planner again.
4. Once it connects, it should report “No Fix” if you're indoors or don't have a fix yet or “3D Fix” if you've got GPS fix. Either is great, it means your new GPS was detected!

Montgomery County
Community College