# Lesson 1: Terminology

http://www.robotshop.com/blog/en/make-uav-lesson-1-platform-rtf-arf-kit-custom-13989

# Lesson 2: The Frame

http://www.robotshop.com/blog/en/make-uav-lesson-2-platform-14448

# Lesson 3: Propulsion

http://www.robotshop.com/blog/en/make-uav-lesson-3-propulsion-14785

# Lesson 4: Flight Controller

* You should have:
  + Your UAV frame
  + Motors, Propellers, ESCs, Battery
* A flight controller is an integrated circuit with a microprocessor, sensors, and input/output pins
* Out of the box, the controller does not magically know your UAV type or configuration
  + So you need to set certain parameters in a software program and have it uploaded to the board

## Main Processor

**8051 vs. AVR vs. PIC vs. ARM:** These microcontroller families are the basis of most current flight controllers.

* Arduino is AVR based
* Microchip is the primary manufacturer of PIC chips
* ARM (STM32 for example) uses 16/32-bit architecture
* AVR and PIC tend to use 8/16-bit
* As boards become less expensive, expect to see new generations of flight controllers which can run full operating systems like Linux and Android

**CPU:** Usually work in multiples of 8-bit (8-bit, 16-bit, 32-bit, 64-bit), and **is a reference to the primary registers in a CPU**

* Microprocessors can only process a set (maximum) number of bits at a time
* The more bits it can handle, the more accurate (and faster) the processing will be
* E.g. processing a 16-bit variable on an 8-bit processor might be slow, but very fast on a 32-bit processor
* The code also needs to work for the right number of bits, few programs are optimized for 32 bits

**Operating frequency:** Frequency at which the main processor operates, measured in Hertz

* Basically, the **clock rate**

**Program memory / Flash:** Where the main code is stored

* Useful when storing in flight data like GPS coordinates, flight plans, automated camera movement, etc.
* The code uploaded remains on the chip even if the power is cut

**SRAM:** “Static Random-Access Memory”, the space on the chip used when making calculations

* Data stored in RAM is lost when power is cut
* Higher RAM means more information is “readily available” for calculations at any given time

**EEPROM:** “Electrically Erasable Programmable Read-Only Memory”, used to store information which does not change in flight, like settings

* Unlike SRAM which can relate to sensor data, etc.

**Additional I/O Pins:** Digital and Analogue pins, some are used by the sensors, others for communication, and others just for general input and output

* Can be connected to RC servos

**A/D Converter:** Analogue to digital converted. Used if the sensors onboard output analoge voltages (0-3.3V or 0-5V) and translates them to digital data

* Just like the CPU, the maximum number of bits processed by the A/D converter determines accuracy
* Related to this is the frequency at which the microprocessor can read data (per second) to ensure no information is lost
* The higher the A/D conversion, the more accurate, but it is important the the converter can handle the rate at which information is sent

## Power

There are two voltage ranges usually specified for a flight controller:

* The voltage input range of the flight controller (usually ~5V)
* The voltage input range of the microprocessor’s logic (3.3V or 5V)

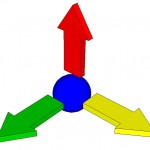
Usually you only pay attention to the flight controller input, it takes care of the microprocessor. Most operate at 5V because that is the voltage provided by the BEC

* **You should ideally not power the flight controller separately from the main battery**
* Unless you want a battery backup in case the flgith controller draws more than 5V (or whatever voltage), which the BEC can’t provide. This would cause a brownout/reset.
* Usually a capacitor is used rather than a battery backup

## Sensors

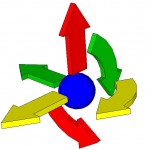
A flight controller is just a programmable microcontroller, but with specific sensors on board. At a bare minimum, the flight controller will just have three axis gyroscope but not be able to auto-level. Flight controllers will usually have some of the following:

**Accelerometer**

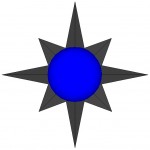
* Measure linear acceleration in up to three axes
* Usually given in gs (units of gravity)
* Can be integrated twice to give position, but is subject to “drift” because of losses in the output
* Because they detect gravity, they can tell you which direction is “down”, which plays a major role in keeping rotorcraft stable
* Should be mounted to the flight controller so that the linear axes line up with the main axes of the UAV

**Gyroscope**

* Measures the rate of angular change in up to three angular axes (say alpha, beta and gamma)
  + Usually in Degrees per second
* It does not measure absolute angles but you can also integrate it to get orientation, which is also subject to “drift”
* Output of the gyroscope is usually analog or I2C, but you don’t usually worry about this because it’s handled by the flight controller’s code
* Should also be mounted so axes are lined up with UAV

**Inertial Measurement Unit (IMU)**

* Just a small board that contains both an accelerometer and a gyroscope
  + Both of which would be three-axis
* May also contain additional sensors such as three axis magnetometer, for a total of 9 axes measurement

**Compass / Magnetometer**

* Electronic magnetic compass able to measure the earth’s magnetic field and uses it to determine the drone’s compass direction with respect to magnetic north
* Almost always present if the system has a GPS input
* Is available in one to three axis

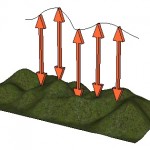
**Pressure / Barometer**

* Can give pretty accurate measure of UAV’s height based on pressure
* Most flight contrallers take input from both pressure sensor and GPS altitude to calculate accurate height above sea level
* It is preferable to have the barometer covered with a piece of foam to diminish the effects of wind over the chip

**GPS**

* Global positioning systems use the signals from a number of satellites in orbit around the earth to determine their specific geographic location
* A flight controller may have an onboard GPS or one connected by a cable
* The GPS antenna should not be confused with the GPS chip itself, and can look like a small black box or a normal “duck” antenna
* More accurate if it gets signals from more satellites

**Distance**

* Used because GPS and pressure sensors alone can’t tell you how far away from the ground you are
* Usually a downward facing ultrasonic, laser or lidar technology sensor (infrared has issues in sunlight)
* Very few flight controllers include distance sensors as part of the package

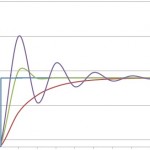
## Flight Modes

Not all flight modes below are available on all flight controllers. A “flight mode” is the way a flight controller uses sensors and RC input to fly and stabilize an aircraft. If you have a transmitter with 5 or more channels, you can configure the 5+ channels to swith between those modes in flight.

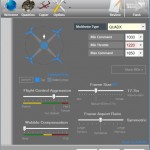
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Mode** | **Gyro** | **Accel** | **Barom** | **Comp** | **GPS** | **Notes** |
| **ACRO** / Gyro Only | Y |  |  |  |  | Default mode, for “acrobatic” flight (drone does not auto-level) |
| **Angle** (Stable/Level/Acc) | Y | Y |  |  |  | Stable mode; keeps drone level above ground but not at specific position |
| **Horizon** | Y |  |  |  |  | “Combines stable effects with slow RC commands and acrobatics with fast RC commands |
| **BARO** (Altitude Hold) | Y | Y | Y |  |  | Maintains a certain height when no other commands are received |
| **MAG** (Heading Hold) | Y | Y |  | Y |  | Compass direction lock mode, will try to keep the yaw orientation fixed |
| **HEADFREE** (CareFree) | Y | Y |  | Y |  | Holds orientation (yaw) of drone and always moves in same 2D direction for same ROLL/PITCH stick movement |
| **GPS** / Return to Home | Y | Y |  | Y | Y | Uses compass and GPS to return to the starting GPS point |
| **GPS** / Waypoint | Y | Y |  | Y | Y | Follows pre-configured GPS way-points autonomously |
| **GPS** / Position Hold | Y | Y |  | Y | Y | Hold current position using GPS and Barometer (if available) |
| **Failsafe** | Y |  |  |  |  | Aircraft reverts to ACRO/Gyro when no other modes are selected |

## Software

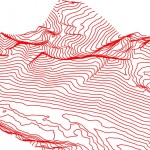
**PID Control Loop & Tuning**

* Proportional Integral Derivative (PID) control allows to change a drone’s flight characteristics like:
  + How it reacts to input
  + How well and quickly it stabilizes
* Manufacturers which produce “ready to fly” (RTF) kits are able to fine tune the PID settings and equations for their specific platforms
  + Which is why most RTF multi-rotors fly well out of the box
* If you build your custom drone however, you need a flight controller suitable to almost any type of multi-rotor aircraft
  + So it’s up to the end-user to adjust the values until they are satisfied with the flight characteristics

**GUI**

* Used to visually edit the code which will be uploaded to the flight controller
* They used to be text-based interfaces which required that you understand almost all of the code and change specific settings
* More recently, they have become interactive GUIs with just a few adjustable parameters

**Additional Features**

* Some flight controllers may have software features not on others, which include
  + Autonomous waypoint navigation: you can set GPS waypoints which the drone will follow autonomously
  + “Orbiting”, i.e. moving around a fixed GPS coordinate with the front of the drone always pointed towards a specific coordinate (useful for folming)
  + “Follow me”, usually GPS based, can allow tracking based on GPS coordinates (of a smartphone for example)
  + 3D imaging: usually done after the flight using pictures captured during the flight and GPS data
  + Other open source stuff

## Communication

**Radio Control (RC)**

* Usually involves hand-held RC transmitter and RC receiver
* For UAVs, you need a minimum of 4 channels though more are suggested, even if not used, the 4 are used for:
  + Pitch (forward / backward motion)
  + Elevation (closer or farther from the ground)
  + Yaw (rotating clockwise or counter-clockwise)
  + Roll (to strafe left and right)
* Additional channels can be used for:
  + **Arming** / Disarming the motors
  + **Gimbal controls** (pan up/down, rotate CW / CCW, zoom)
  + **Change flight modes**
  + **Activate/deploy** a payload, parachute, buzzer, other device
  + Any number of uses
* On it’s own, the receiver simplyrelays the value from the transmitter, so it cannot control the UAV alone
* The receiver must be connected to the flight controller, which must be programmed to access RC signals
  + Most flight controllers on the market accept RC signals right away from a receiver and even power the receiver from one of their pins
* Additional considerations include:
  + Not all RC transmitters can provide the full RC signal range of 500ms to 2500ms; some artificially limit this since most RC applications are for RC cars, airplanes and helicopters
  + The range of the RC system is almost never provided because it involves many factors like obstructions, battery charge, temperature, humidity, etc.
  + Some RC systems have a receiver that also has a built-in transmitter for transmitting sensor data to the RC transmitter’s display
    - For things like GPS coordinates

**Bluetooth**

* Originally meant to allow transfer of data between devices without worrying about matching frequencies
* Some controllers can send and receive data via Bluetooth

**WiFi**

* Can handle both data transmission and image transmission, but is much more difficult to set up/implement

**Radio Frequency (FC)**

* Sending data from a computer or microcontroller wirelessly to the drone using and RF transmitter/receiver
* Using a normal RF unit connected to a computer allows for long range two-way communication with a high “density” of data

**SmartPhone**

* Usually include Bluetooth and/or WiFi, either of which can be used to control the drone or receive data/video

**Infrared (IR)**

* Not reliable because there’s IR interference everywhere

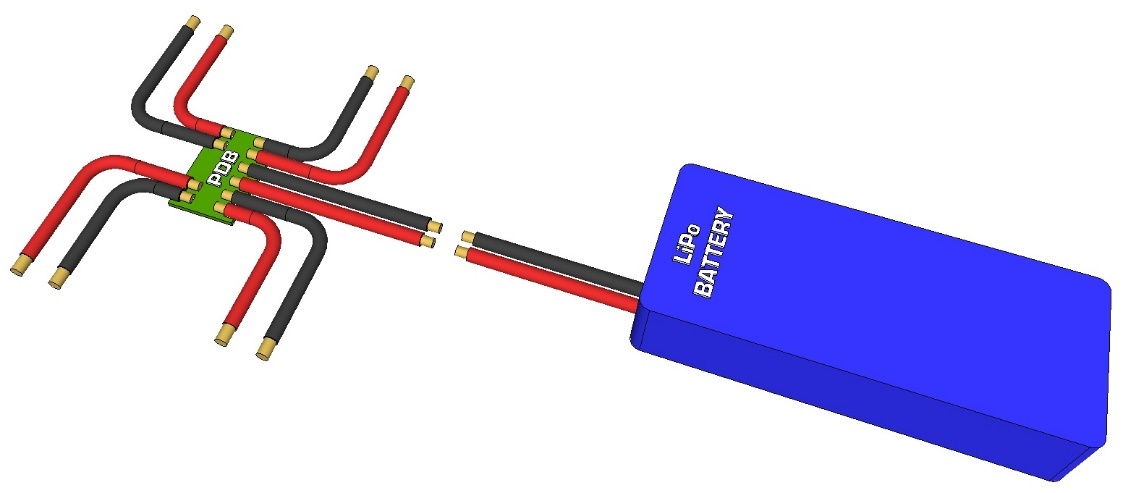
# Lesson 5: Assembly

At this point you should have:

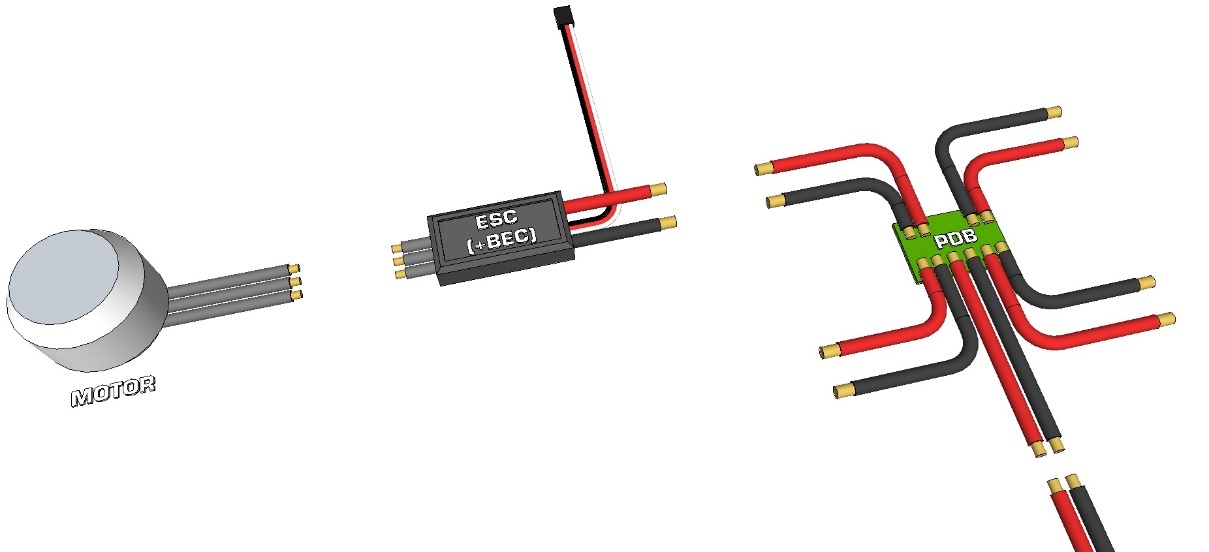
* Frame
* Motors, ESCs, Propellers, Battery Charger
* Power Distribution Board / Harness
* Flight Controller & Communication Device (RC suggested)

## Propulsion System

### Battery -> Power Distribution

* If they have the same connector, then just connect them and you’re done
* If they don’t, you might need an adapter between the battery and power distribution
  + DO NOT CUT THE WIRES TO REMOVE THE CONNECTOR FROM THE BATTERY
* Alternatively, look for the mating connector to that of your battery and order it, cut of the connector from your power distribution system and solder the new one

### Motor -> ESC -> Power Distribution

* The Power Distribution Board (PDB) splits the main battery to each of the ESCs
  + It needs to split the main battery into the number of motors (and thus ESCs) that you have
* The power is provided “as is” to the ESCs, there is no step up or step down voltage
* An ESC has the following wires:
  + One 3-pin, 0.1” spaced R/C connector, the balck pin is GND, the rered pin is 5V (via BEC) and the yellow/white one is the signal IN
  + Three separate wires to be connected to the brushless DC motor (I guess order doesn’t matter…?)
  + Two battery input connectors to plug into the PDB
* ESCs normally have a built in BEC to convert the battery’s voltage to 5V to be used by the receiver and flight controller
  + The 5V is provided via the RC connector from the ESC
  + The red pin is usually the 5V output on the RC connectors
  + **You only need one BEC to power the flight controller**
  + **THE ESCs, ONCE CONNECTED TO THE BATTERY, ARE WHAT GENERALLY POWERS THE FLIGHT CONTROLLER**