Battery Monitoring

Betaflight has a battery monitoring feature. The voltage of the main battery can be measured by the system and used to trigger a low-battery warning buzzer (Buzzer.md), on-board status LED flashing and LED strip patterns.

Low battery warnings can:

- Help ensure you have time to safely land the aircraft
- Help maintain the life and safety of your LiPo/LiFe batteries, which should not be discharged below manufacturer recommendations

Minimum and maximum cell voltages can be set, and these voltages are used to auto-detect the number of cells in the battery when it is first connected.

Per-cell monitoring is not supported, as we only use one ADC to read the battery voltage.

Supported targets

All targets support battery voltage monitoring unless status.

Connections

When dealing with batteries **ALWAYS CHECK POLARITY!**

Measure expected voltages **first** and then connect to the flight controller. Powering the flight controller with
incorrect voltage or reversed polarity will likely fry your flight controller. Ensure your flight controller
has a voltage divider capable of measuring your particular battery voltage.

Naze32

The Naze32 has an on-board battery divider circuit; just connect your main battery to the VBAT connector.

CAUTION: When installing the connection from main battery to the VBAT connector, be sure to first disconnect the main battery from the frame/power distribution board. Check the wiring very carefully before connecting battery again. Incorrect connections can immediately and completely destroy the flight controller and connected peripherals (ESC, GPS, Receiver etc.).

CC3D

The CC3D has no battery divider. To use voltage monitoring, you must create a divider that gives a 3.3v
MAXIMUM output when the main battery is fully charged. Connect the divider output to S5_IN/PA0/RC5.

Notes:

- S5_IN/PA0/RC5 is Pin 7 on the 8 pin connector, second to last pin, on the opposite end from the
  GND/+5/PPM signal input.
- When battery monitoring is enabled on the CC3D, RC5 can no-longer be used for PWM input.

Sparky

See the Sparky board chapter (boards/Board%20-%20Sparky.md).

Configuration

Enable the VBAT feature.

Configure min/max cell voltages using the following CLI setting:

`vbat_scale` - Adjust this to match actual measured battery voltage to reported value (which may be displayed via the `status` command)

`vbat_max_cell_voltage` - Maximum voltage per cell, used for auto-detecting battery voltage in 0.1V units, i.e. 43 = 4.3V

`vbat_min_cell_voltage` - Minimum voltage per cell; this triggers battery-critical alarms, in 0.1V units, i.e. 33 = 3.3V

`vbat_warning_cell_voltage` - Warning voltage per cell; this triggers battery-warning alarms, in 0.1V units, i.e. 34 = 3.4V

`vbat_hysteresis` - Sets the hysteresis value for low-battery alarms, in 0.1V units, i.e. 1 = 0.1V

e.g.

```
set vbat_scale = 110
set vbat_max_cell_voltage = 43
set vbat_min_cell_voltage = 33
set vbat_warning_cell_voltage = 34
set vbat_hysteresis = 1
```

Current Monitoring

Current monitoring (amperage) is supported by connecting a current meter to the appropriate current meter ADC input (see the documentation for your particular board).

When enabled, the following values calculated and used by the telemetry and OLED display subsystems:

- Amps
- mAh used
- Capacity remaining

Configuration

Enable current monitoring using the CLI command:

```
feature CURRENT_METER
```

Configure the current meter type using the `amperage_meter_type` settings here:

Value	Sensor Type
NONE	None
ADC	ADC/hardware sensor
VIRTUAL	Virtual sensor

Configure capacity using the `battery_capacity` setting, in mAh units.

If you're using an OSD that expects the multiwii current meter output value, then set `multiwii_amperage_meter_output` to `ON` (this multiplies amperage sent to MSP by 10 and truncates negative values)).

ADC Sensor

The current meter needs to be configured so the value read at the Analog to Digital Converter (ADC) input matches actual current draw. Just like you need a voltmeter to correctly calibrate your voltage reading you also need an ammeter to calibrate the current sensor.
Unlike voltage sensing which is usually quite good from the factory, current sensing varies significantly from board to board and should be calibrated.

It is recommended to set `multiwii_amperage_meter_output` to `OFF` when calibrating ADC current sensor.

To measure the current a linear response device is used that converts the current traveling through it to a voltage to be read by the ADC on your flight controller.
The maximum voltage that the flight controller can read is 3.3V (3300 mV), this is usually the limiting factor on the maximum measurable current.
Most sensors use a shunt resistor to measure current however there are a few that use hall effect sensors.

The flight controller uses the following equation to convert the measured ADC voltage to a current.

```
Current(Amps) = ADC (mV) / amperage_meter_scale * 10 + amperage_meter_offset / 1000
```

Where the calibrations are:

Setting	Description
`amperage_meter_scale`	The scaling factor in mV/10A
`amperage_meter_offset`	The offset in mA

This is in the mathematical form of y = x/m + b and with a few measurements along this line you can calibrate any combination of sensor and flight controller to a high accuracy.

Calibrate using an ammeter

!!Important: Always take off your props before doing any testing!!

To calibrate your flight controller with a current meter follow these steps.

1. Make a copy of this google sheet (https://docs.google.com/spreadsheets/d/1lkL-X_FT9x2oqrwQEctDsEUhgdY19upNGc78M6FfJXY/). It will do all of the maths for you.
2. Hook your ammeter up in series with your drone and a charged battery. I suggest an XT60 extender with one lead cut. Now your ammeter will be displaying the true current draw of your system.
3. Connect to your flight controller through the configurator and check your current calibrations. Change them in the google sheet if needed.
4. Use the motor tab to increase the throttle and change the current draw of the drone to around 1 A on the ammeter (it does not matter if it is not exact).
5. Switch back to the power and battery tab and record current from the ammeter in the measured current column and the current reported by Betaflight in the flight controller current column (both in amps, to 2 decimal places).
6. Repeat this measurement (steps 4 and 5) 3 or more times at various currents from 0 to 5 Amps (make sure not to go over your ammeter rated current).
7. Once this is done make sure the results are linear on the graph and that the regression value is green. You can now update to the new calibration values and enjoy accurate battery usage information.

The same method can be applied to both hall effect sensors and shunt resistors. Shunt resistors will usually have an offset of less than +/- 1000 mA however hall effect sensors offsets will be much higher.
Note that while your calibration may be correct there is still a maximum current measure that you may exceed at maximum throttle, this will cause the current recorded by the flight controller to ceiling at this value even if the actual current is higher.
As a result the reported mAh used may be less than is actually used, always keep an eye on the battery voltage as well.

If you do not want to use google sheets then simply use some other tool that preforms a linear regression on the dataset. Multiply `amperage_meter_scale` used in testing by the slope and subtract the intercept in mA from `amperage_meter_offset` to get the corrected calibration values.

Virtual Sensor

The virtual sensor uses the throttle position to calculate an estimated current value. This is useful when a real sensor is not available. The following settings adjust the virtual sensor calibration:

Setting	Description
`amperage_meter_scale`	The throttle scaling factor [centiamps, i.e. 1/100th A]
`amperage_meter_offset`	The current at zero throttle (while disarmed) [centiamps, i.e. 1/100th A]

There are two simple methods to tune these parameters: one uses a battery charger and another depends on actual current measurements.

Tuning Using Actual Current Measurements

If you know your craft's current draw (in Amperes) while disarmed (Imin) and at maximum throttle while armed (Imax), calculate the scaling factors as follows:

```
amperage_meter_scale = (Imax - Imin) * 100000 / (Tmax + (Tmax * Tmax / 50))
amperage_meter_offset = Imin * 100
```

Note: Tmax is maximum throttle offset (i.e. for `max_throttle` = 1850, Tmax = 1850 - 1000 = 850)

For example, assuming a maximum current of 34.2A, a minimum current of 2.8A, and a Tmax `max_throttle` = 1850:

```
amperage_meter_scale = (Imax - Imin) * 100000 / (Tmax + (Tmax * Tmax / 50))
                     = (34.2 - 2.8) * 100000 / (850 + (850 * 850 / 50))
                     = 205
amperage_meter_offset = Imin * 100 = 280
```

Tuning Using Battery Charger Measurement

If you cannot measure current draw directly, you can approximate it indirectly using your battery charger.
However, note it may be difficult to adjust `amperage_meter_offset` using this method unless you can
measure the actual current draw with the craft disarmed.

Note:

- This method depends on the accuracy of your battery charger; results may vary.
- If you add or replace equipment that changes the in-flight current draw (e.g. video transmitter,
  camera, gimbal, motors, prop pitch/sizes, ESCs, etc.), you should recalibrate.

The general method is:

1. Fully charge your flight battery
2. Fly your craft, using >50% of your battery pack capacity (estimated)
3. Note Cleanflight's reported mAh drawn
4. Fully charge your flight battery again, noting the amount of mAh recharged
5. Adjust `amperage_meter_scale` to according to the formula given below
6. Repeat and test

Given (a) the mAh recharged and (b) the cleanflight reported mAh drawn, calculate a new `amperage_meter_scale` value as follows:

```
amperage_meter_scale = old_amperage_meter_scale * (mAh_recharged /
cleanflight_reported_mAh_drawn)
```

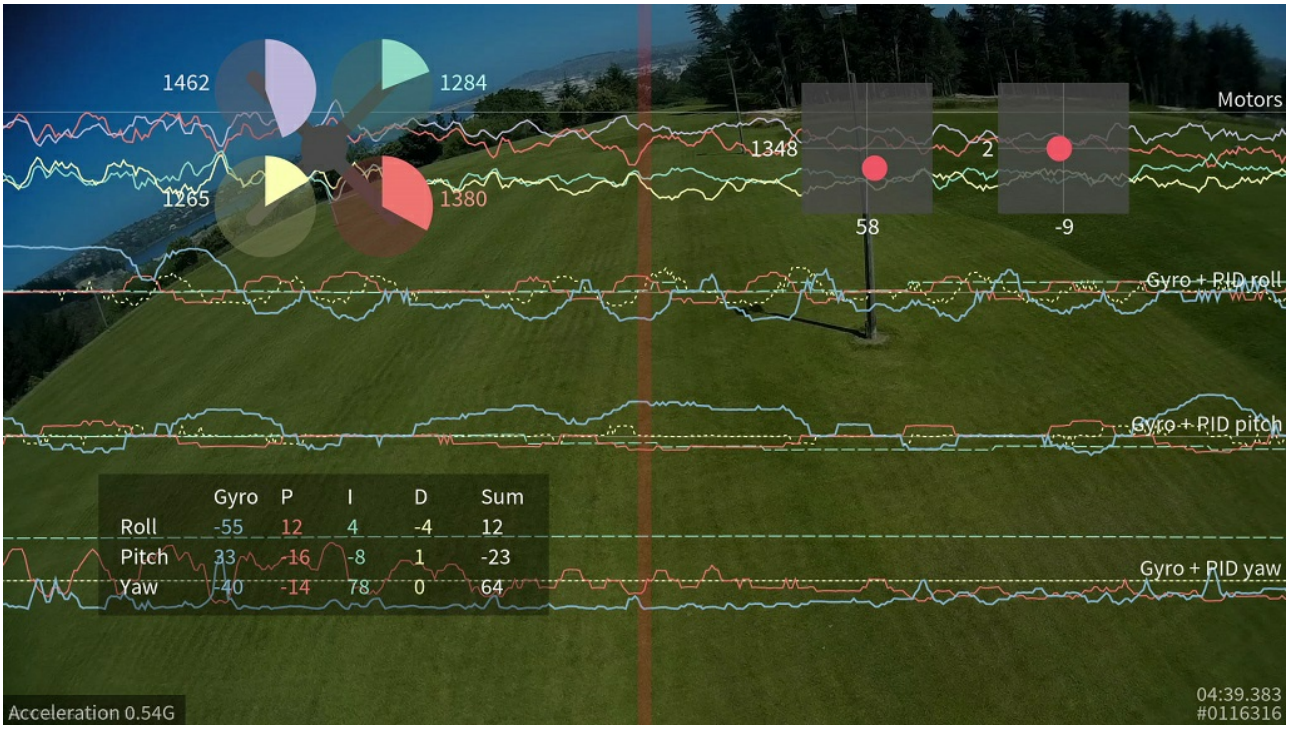
For example, assuming:

- An amount recharged of 1500 mAh
- A Cleanflight reported current drawn of 2000 mAh
- An existing `amperage_meter_scale` value of 400 (the default)

Then the updated `amperage_meter_scale` is:

```
amperage_meter_scale = old_amperage_meter_scale * (mAh_recharged /
cleanflight_reported_mAh_drawn)
                     = 400 * (1500 / 2000)
                     = 300
```

Blackbox flight data recorder



Introduction

This feature transmits your flight data information on every control loop iteration over a serial port to an external
logging device like an OpenLog to be recorded, to an onboard dataflash chip which is present on some flight controllers,
or to an onboard SD card socket.

After your flight, you can view the resulting logs using the interactive log viewer:

https://github.com/cleanflight/blackbox-log-viewer

You can also use the `blackbox_decode` tool to turn the logs into CSV files for analysis, or render your flight log as a
video using the `blackbox_render` tool. Those tools can be found in this repository:

https://github.com/cleanflight/blackbox-tools

Logged data

The blackbox records flight data on every iteration of the flight control loop. It records the current time in
microseconds, P, I and D corrections for each axis, your RC command stick positions (after applying expo curves),
gyroscope data, accelerometer data (after your configured low-pass filtering), barometer and sonar readings, 3-axis
magnetometer readings, raw VBAT and current measurements, RSSI, and the command being sent to each motor speed

controller. This is all stored without any approximation or loss of precision, so even quite subtle problems should be
detectable from the fight data log.

GPS data is logged whenever new GPS data is available. Although the CSV decoder will decode this data, the video
renderer does not yet show any of the GPS information (this will be added later).

Supported configurations

The maximum data rate that can be recorded to the flight log is fairly restricted, so anything that increases the load
can cause the flight log to drop frames and contain errors.

The Blackbox is typically used on tricopters and quadcopters. Although it will work on hexacopters and octocopters,
because these craft have more motors to record, they must transmit more data to the flight log. This can increase the
number of dropped frames. Although the browser-based log viewer supports hexacopters and octocopters, the command-line
`blackbox_render` tool currently only supports tri- and quadcopters.

Cleanflight's `looptime` setting decides how frequently an update is saved to the flight log. The default looptime on
Cleanflight is 3500. If you're using a looptime smaller than about 2400, you may experience some dropped frames due to
the high required data rate. In that case you will need to reduce the sampling rate in the Blackbox settings, or
increase your logger's baudrate to 250000. See the later section on configuring the Blackbox feature for details.

Setting up logging

First, you must enable the Blackbox feature. In the [Cleanflight Configurator (https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en)](https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en) enter the Configuration tab,
tick the "BLACKBOX" feature at the bottom of the page, and click "Save and reboot"

Now you must decide which device to store your flight logs on. You can either transmit the log data over a serial port
to an external logging device like the [OpenLog serial data logger (https://www.sparkfun.com/products/9530)](https://www.sparkfun.com/products/9530) to be recorded to a microSDHC card, or if you have a compatible flight controller you can store the logs on the onboard dataflash storage instead.

OpenLog serial data logger

The OpenLog is a small logging device which attaches to your flight controller using a serial port and logs your
flights to a MicroSD card.

The OpenLog ships from SparkFun with standard "OpenLog 3" firmware installed. Although this original OpenLog firmware
will work with the Blackbox, in order to reduce the number of dropped frames it should be

reflashed with the
higher performance OpenLog Blackbox firmware (https://github.com/cleanflight/blackbox-firmware). The special Blackbox variant of the OpenLog firmware also ensures that
the OpenLog is using Cleanflight compatible settings, and defaults to 115200 baud.

You can find the Blackbox version of the OpenLog firmware here (https://github.com/cleanflight/blackbox-firmware),
along with instructions for installing it onto your OpenLog.

microSDHC

Your choice of microSDHC card is very important to the performance of the system. The OpenLog relies on being able to
make many small writes to the card with minimal delay, which not every card is good at. A faster SD-card speed rating is
not a guarantee of better performance.

microSDHC cards known to have poor performance

- Generic 4GB Class 4 microSDHC card - the rate of missing frames is about 1%, and is concentrated around the most
  interesting parts of the log!
- Sandisk Ultra 32GB (unlike the smaller 16GB version, this version has poor write latency)

microSDHC cards known to have good performance

- Transcend 16GB Class 10 UHS-I microSDHC (typical error rate < 0.1%)
- Sandisk Extreme 16GB Class 10 UHS-I microSDHC (typical error rate < 0.1%)
- Sandisk Ultra 16GB (it performs only half as well as the Extreme in theory, but still very good)

You should format any card you use with the SD Association's special formatting tool (https://www.sdcard.org/downloads/formatter_4/) , as it will give the OpenLog
the best chance of writing at high speed. You must format it with either FAT, or with FAT32 (recommended).

Choosing a serial port for the OpenLog

First, tell the Blackbox to log using a serial port (rather than to an onboard dataflash chip). Go to the
Configurator's CLI tab, enter `set blackbox_device=SERIAL` to switch logging to serial, and save.

You need to let Cleanflight know which of your serial ports (https://github.com/cleanflight/cleanflight/blob/master/docs/Serial.md) you connect your OpenLog to (i.e. the Blackbox port),
which you can do on the Configurator's Ports tab.

You should use a hardware serial port (such as UART1 on the Naze32, the two-pin Tx/Rx header in the center of the
board). SoftSerial ports can be used for the Blackbox. However, because they are limited to

19200 baud, your logging
rate will need to be severely reduced to compensate. Therefore the use of SoftSerial is not recommended.

When using a hardware serial port, Blackbox should be set to at least 115200 baud on that port. When using fast
looptimes (<2500), a baud rate of 250000 should be used instead in order to reduce dropped frames.

The serial port used for Blackbox cannot be shared with any other function (e.g. GPS, telemetry) except the MSP
protocol. If MSP is used on the same port as Blackbox, then MSP will be active when the board is disarmed, and Blackbox
will be active when the board is armed. This will mean that you can't use the Configurator or any other function that
requires MSP, such as an OSD or a Bluetooth wireless configuration app, while the board is armed.

Connect the "TX" pin of the serial port you've chosen to the OpenLog's "RXI" pin. Don't connect the serial port's RX
pin to the OpenLog, as this will cause the OpenLog to interfere with any shared functions on the serial port while
disarmed.

Naze32 serial port choices

On the Naze32, the TX/RX pins on top of the board are connected to UART1, and are shared with the USB connector.
Therefore, MSP must be enabled on UART1 in order to use the Configurator over USB. If Blackbox is connected to the pins
on top of the Naze32, the Configurator will stop working once the board is armed. This configuration is usually a good
choice if you don't already have an OSD installed which is using those pins while armed, and aren't using the FrSky
telemetry pins.

Pin RC3 on the side of the board is UART2's Tx pin. If Blackbox is configured on UART2, MSP can still be used on UART1
when the board is armed, which means that the Configurator will continue to work simultaneously with Blackbox logging.
Note that in `PARALLEL_PWM` mode this leaves the board with 6 input channels as RC3 and RC4 pins are used by UART2 as Tx and Rx. Cleanflight automatically shifts logical channel mapping for you when UART2 is enabled in `Ports` tab so you'll have to shift receiver pins that are connected to Naze32 pins 3 to 6 by two.

The OpenLog tolerates a power supply of between 3.3V and 12V. If you are powering your Naze32 with a standard 5V BEC,
then you can use a spare motor header's +5V and GND pins to power the OpenLog with.

Other flight controller hardware

Boards other than the Naze32 may have more accessible hardware serial devices, in which case refer to their
documentation to decide how to wire up the logger. The key criteria are:

- Should be a hardware serial port rather than SoftSerial.
- Cannot be shared with any other function (GPS, telemetry) except MSP.
- If MSP is used on the same UART, MSP will stop working when the board is armed.

OpenLog configuration

Power up the OpenLog with a microSD card inside, wait 10 seconds or so, then power it down and plug the microSD card
into your computer. You should find a "CONFIG.TXT" file on the card, open it up in a text editor. You should see the
baud rate that the OpenLog has been configured for (usually 115200 or 9600 from the factory). Set the baud rate to match
the rate you entered for the Blackbox in the Configurator's Port tab (typically 115200 or 250000).

Save the file and put the card back into your OpenLog, it will use those settings from now on.

If your OpenLog didn't write a CONFIG.TXT file, create a CONFIG.TXT file with these contents and store it in the root
of the MicroSD card:

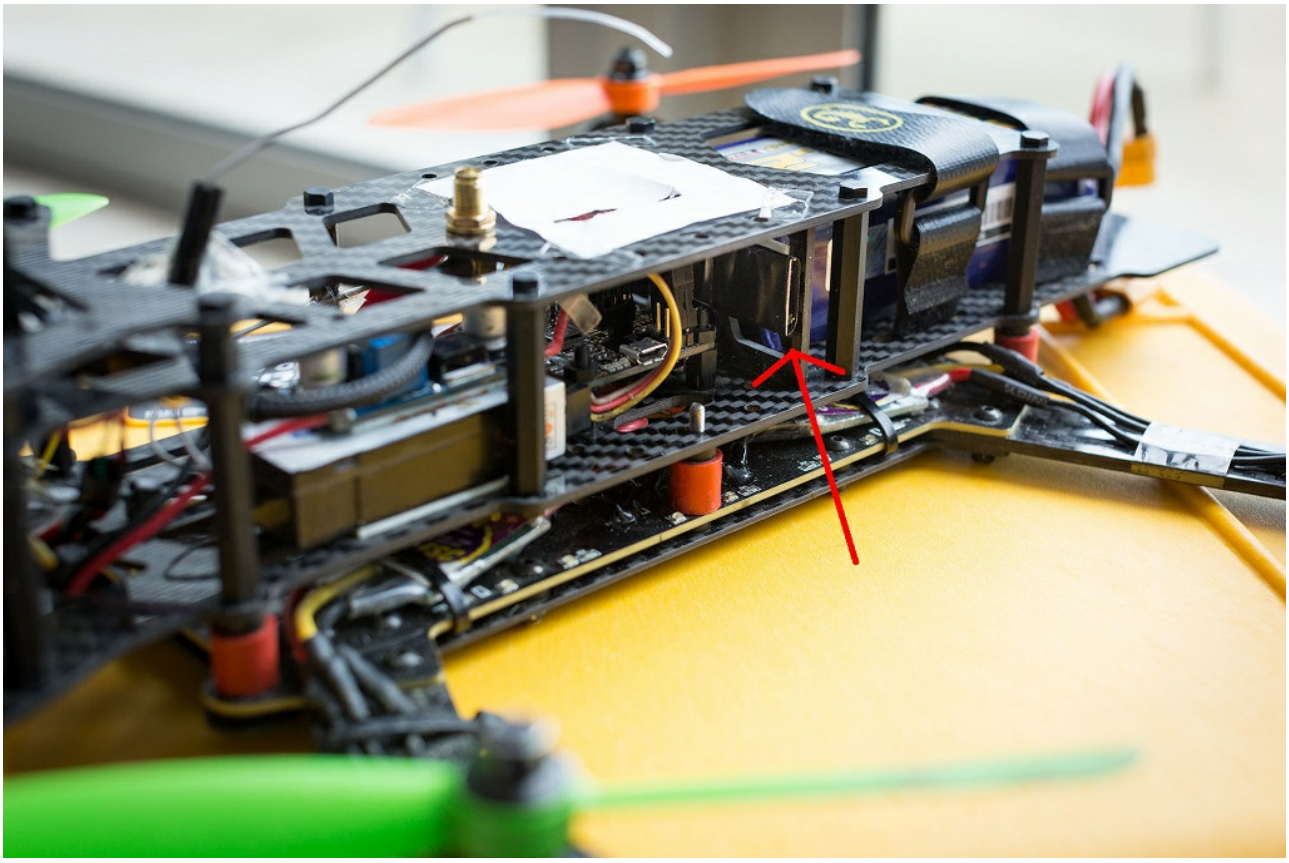
```
115200
baud
```

If you are using the original OpenLog firmware, use this configuration instead:

```
115200,26,0,0,1,0,1
baud,escape,esc#,mode,verb,echo,ignoreRX
```

OpenLog protection

The OpenLog can be wrapped in black electrical tape or heat-shrink in order to insulate it from conductive frames (like
carbon fiber), but this makes its status LEDs impossible to see. I recommend wrapping it with some clear heatshrink
tubing instead.



Onboard dataflash storage

Some flight controllers have an onboard SPI NOR dataflash chip which can be used to store flight logs instead of using
an OpenLog.

The full version of the Naze32 and the CC3D have an onboard "m25p16" 2 megabyte dataflash storage chip. This is a small
chip with 8 fat legs, which can be found at the base of the Naze32's direction arrow. This chip is not present on the
"Acro" version of the Naze32.

The SPRacingF3 has a larger 8 megabyte dataflash chip onboard which allows for longer recording times.

These chips are also supported:

- Micron/ST M25P16 - 16 Mbit / 2 MByte (datasheet (http://www.micron.com/~/media/Documents/Products/Data%20Sheet/NOR%20Flash/Se
- Micron/ST N25Q064 - 64 Mbit / 8 MByte (datasheet (http://www.micron.com/~/media/documents/products/data-sheet/nor-flash/serial-nor/n25q/n25q_64a_3v_65nm.pdf))
- Winbond W25Q64 - 64 Mbit / 8 MByte (datasheet (http://www.winbond.com/resource-files/w25q64fv_revl1_100713.pdf))
- Macronix MX25L64 - 64 Mbit / 8 MByte (datasheet (http://media.digikey.com/pdf/Data%20Sheets/Macronix/MX25L6406E.pdf))
- Micron/ST N25Q128 - 128 Mbit / 16 MByte (datasheet (http://www.micron.com/~/media/Documents/Products/Data%20Sheet/NOR%20Flash/Se
- Winbond W25Q128 - 128 Mbit / 16 MByte (datasheet

(http://www.winbond.com/resource-files/w25q128fv_revhh1_100913_website1.pdf))

Enable recording to dataflash

On the Configurator's CLI tab, you must enter `set blackbox_device=SPIFLASH` to switch to logging to an onboard dataflash chip,
then save.

Onboard SD card socket

Some flight controllers have an SD or Micro SD card socket on their circuit boards. This allows for very high speed
logging (1KHz or faster, which is a looptime of 1000 or lower) on suitable cards.

The card can be either Standard (SDSC) or High capacity (SDHC), and must be formatted with the FAT16 or FAT32
filesystems. This covers a range of card capacities from 1 to 32GB. Extended capacity cards (SDXC) are not supported.

The first time you power up Cleanflight with a new card inserted, the flight controller will spend a few seconds
scanning the disk for free space and collecting this space together into a file called "FREESPAC.E". During flight,
Cleanflight will carve chunks from this file to create new log files. You must not edit this file on your computer (i.e.
open it in a program and save changes) because this may cause it to become fragmented. Don't run any defragmentation
tools on the card either.

You can delete the FREESPAC.E file if you want to free up space on the card to fit non-Blackbox files (Cleanflight will
recreate the FREESPAC.E file next time it starts, using whatever free space was left over).

The maximum size of the FREESPAC.E file is currently 4GB. Once 4GB worth of logs have been recorded, the FREESPAC.E
file will be nearly empty and no more logs will be able to be recorded. At this point you should either delete the
FREESPAC.E file (and any logs left on the card to free up space), or just reformat the card. A new FREESPAC.E file
will be created by Cleanflight on its next boot.

Enable recording to SD card

On the Configurator's CLI tab, you must enter `set blackbox_device=SDCARD` to switch to logging to an onboard SD card,
then save.

Configuring the Blackbox

The Blackbox currently provides two settings (`blackbox_rate_num` and `blackbox_rate_denom`) that allow you to control
the rate at which data is logged. These two together form a fraction (`blackbox_rate_num / blackbox_rate_denom`) which

decides what portion of the flight controller's control loop iterations should be logged. The default is 1/1 which logs
every iteration.

If you're using a slower MicroSD card, you may need to reduce your logging rate to reduce the number of corrupted
logged frames that `blackbox_decode` complains about. A rate of 1/2 is likely to work for most craft.

You can change the logging rate settings by entering the CLI tab in the [Cleanflight Configurator (https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en)](https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en) and using the `set`
command, like so:

```
set blackbox_rate_num = 1
set blackbox_rate_denom = 2
```

The data rate for my quadcopter using a looptime of 2400 and a rate of 1/1 is about 10.25kB/s. This allows about 18
days of flight logs to fit on my OpenLog's 16GB MicroSD card, which ought to be enough for anybody :).

If you are logging using SoftSerial, you will almost certainly need to reduce your logging rate to 1/32. Even at that
logging rate, looptimes faster than about 1000 cannot be successfully logged.

If you're logging to an onboard dataflash chip instead of an OpenLog, be aware that the 2MB of storage space it offers
is pretty small. At the default 1/1 logging rate, and a 2400 looptime, this is only enough for about 3 minutes of
flight. This could be long enough for you to investigate some flying problem with your craft, but you may want to reduce
the logging rate in order to extend your recording time.

To maximize your recording time, you could drop the rate all the way down to 1/32 (the smallest possible rate) which
would result in a logging rate of about 10-20Hz and about 650 bytes/second of data. At that logging rate, a 2MB
dataflash chip can store around 50 minutes of flight data, though the level of detail is severely reduced and you could
not diagnose flight problems like vibration or PID setting issues.

Usage

The Blackbox starts recording data as soon as you arm your craft, and stops when you disarm.

If your craft has a buzzer attached, you can use Cleanflight's arming beep to synchronize your Blackbox log with your
flight video. Cleanflight's arming beep is a "long, short" pattern. The beginning of the first long beep will be shown
as a blue line in the flight data log, which you can sync against your recorded audio track.

You should wait a few seconds after disarming your craft to allow the Blackbox to finish saving its data.

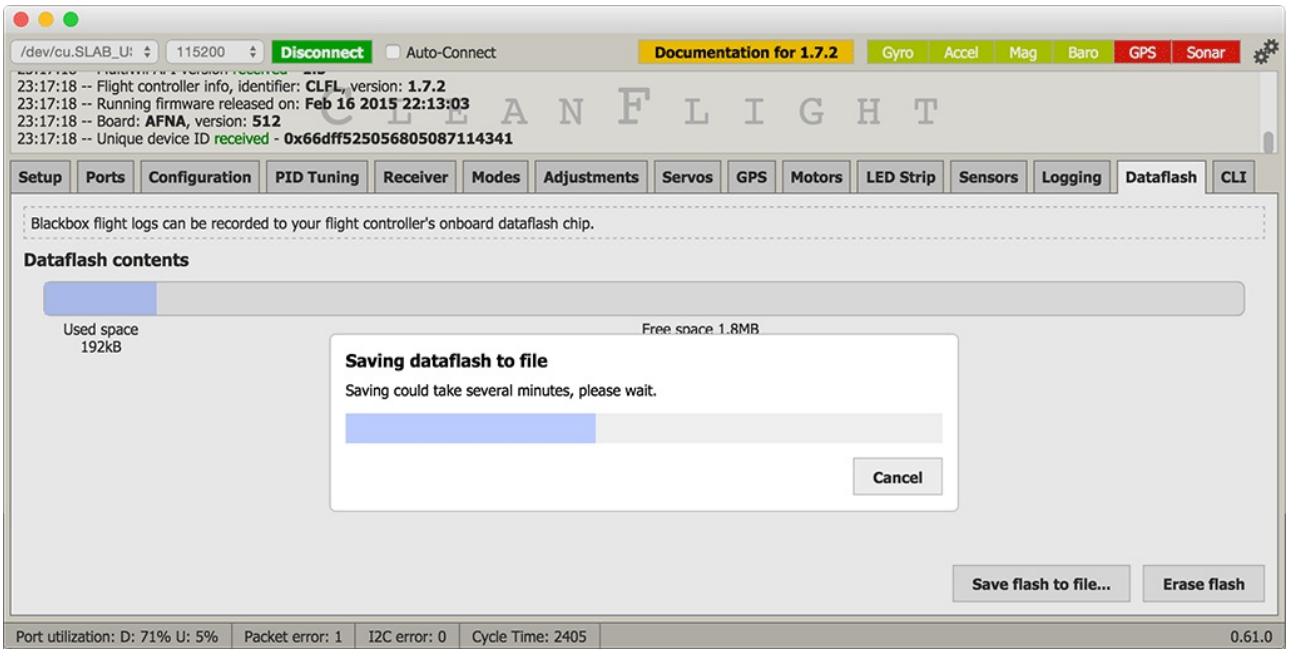
Usage - OpenLog

Each time the OpenLog is power-cycled, it begins a fresh new log file. If you arm and disarm several times without
cycling the power (recording several flights), those logs will be combined together into one file. The command line
tools will ask you to pick which one of these flights you want to display/decode.

Don't insert or remove the SD card while the OpenLog is powered up.

Usage - Dataflash chip

After your flights, you can use the [Cleanflight Configurator (https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en)](https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en) to download the contents of the dataflash to your
computer. Go to the "dataflash" tab and click the "save flash to file..." button. Saving the log can take 2 or 3
minutes.



After downloading the log, be sure to erase the chip to make it ready for reuse by clicking the "erase flash" button.

If you try to start recording a new flight when the dataflash is already full, Blackbox logging will be disabled and
nothing will be recorded.

Usage - Onboard SD card socket

You must insert your SD card before powering on your flight controller. You can remove the SD card while the board is
powered up, but you must wait 5 seconds after disarming before you do so in order to give Cleanflight a chance to finish
saving your log (otherwise the filesystem may become corrupted).

Cleanflight will create a new log file in the "LOG" directory each time the craft is armed. If you are using a Blackbox
logging switch and you keep it paused for the entire flight, the resulting empty log file will be deleted after disarming.

To read your logs, you must remove the SD card and insert it into a card reader on your computer (Cleanflight doesn't
support reading these logs directly through the Configurator).

Usage - Logging switch

If you're recording to an onboard flash chip, you probably want to disable Blackbox recording when not required in order
to save storage space. To do this, you can add a Blackbox flight mode to one of your AUX channels on the Configurator's
modes tab. Once you've added a mode, Blackbox will only log flight data when the mode is active.

A log header will always be recorded at arming time, even if logging is paused. You can freely pause and resume logging
while in flight.

Viewing recorded logs

After your flights, you'll have a series of flight log files with a .TXT extension.

You can view these .TXT flight log files interactively using your web browser with the Cleanflight Blackbox Explorer:

https://github.com/cleanflight/blackbox-log-viewer

This allows you to scroll around a graphed version of your log and examine your log in detail. You can also export a
video of your log to share it with others!

You can decode your logs with the `blackbox_decode` tool to create CSV (comma-separated values) files for analysis,
or render them into a series of PNG frames with `blackbox_render` tool, which you could then convert into a video using
another software package.

You'll find those tools along with instructions for using them in this repository:

https://github.com/cleanflight/blackbox-tools

Buzzer

Cleanflight supports a buzzer which is used for the following purposes:

- Low and critical battery alarms (when battery monitoring enabled)
- Arm/disarm tones (and warning beeps while armed)
- Notification of calibration complete status
- TX-AUX operated beeping - useful for locating your aircraft after a crash
- Failsafe status
- Flight mode change
- Rate profile change (via TX-AUX switch)

If the arm/disarm is via the control stick, holding the stick in the disarm position will sound a repeating tone. This can be used as a lost-model locator.

Three beeps immediately after powering the board means that the gyroscope calibration has completed successfully. Cleanflight calibrates the gyro automatically upon every power-up. It is important that the copter stay still on the ground until the three beeps sound, so that gyro calibration isn't thrown off. If you move the copter significantly during calibration, Cleanflight will detect this, and will automatically re-start the calibration once the copter is still again. This will delay the "three beeps" tone. If you move the copter just a little bit, the gyro calibration may be incorrect, and the copter may not fly correctly. In this case, the gyro calibration can be performed manually via stick command (Controls.md), or you may simply power cycle the board.

There is a special arming tone used if a GPS fix has been attained, and there's a "ready" tone sounded after a GPS fix has been attained (only happens once). The tone sounded via the TX-AUX-switch will count out the number of satellites (if GPS fix).

The CLI command `play_sound` is useful for demonstrating the buzzer tones. Repeatedly entering the command will play the various tones in turn. Entering the command with a numeric-index parameter (see below) will play the associated tone.

Buzzer is enabled by default on platforms that have buzzer connections.

Tone sequences

Buzzer tone sequences (square wave generation) are made so that : 1st, 3rd, 5th, .. are the delays how long the beeper is on and 2nd, 4th, 6th, .. are the delays how long beeper is off. Delays are in milliseconds/10 (i.e., 5 => 50ms).

Sequences available in Cleanflight v1.9 and above are :

```
0    GYRO_CALIBRATED       20, 10, 20, 10, 20, 10   Gyro is calibrated
1    RX_LOST_LANDING       10, 10, 10, 10, 10, 40, 40, 10, 40, 10, 40, 40, 10,
10, 10, 10, 10, 70    SOS morse code
2    RX_LOST               50, 50       TX off or signal lost (repeats until TX
is okay)
3    DISARMING             15, 5, 15, 5     Disarming the board
4    ARMING                30, 5, 5, 5      Arming the board
5    ARMING_GPS_FIX        5, 5, 15, 5, 5, 5, 15, 30    Arming and GPS has fix
6    BAT_CRIT_LOW          50, 2        Battery is critically low (repeats)
7    BAT_LOW               25, 50       Battery is getting low (repeats)
8    NULL                  multi beeps      GPS status (sat count)
9    RX_SET                10, 10       RX is set (when aux channel is set for
beep or beep sequence how many satellites has found if GPS enabled)
10   ACC_CALIBRATION       5, 5, 5, 5       ACC inflight calibration completed
11   ACC_CALIBRATION_FAIL  20, 15, 35, 5    ACC inflight calibration failed
12   READY_BEEP            4, 5, 4, 5, 8, 5, 15, 5, 8, 5, 4, 5, 4, 5    GPS
locked and copter ready
13   NULL                  multi beeps      Variable # of beeps (confirmation,
GPS sat count, etc)
14   DISARM_REPEAT         0, 100, 10       Stick held in disarm position
(after pause)
15   ARMED                 0, 245, 10, 5    Board is armed (after pause ;
repeats until board is disarmed or throttle is increased)
```

Types of buzzer supported

The buzzers are enabled/disabled by simply enabling or disabling a GPIO output pin on the board.
This means the buzzer must be able to generate its own tone simply by having power applied to it.

Buzzers that need an analog or PWM signal do not work and will make clicking noises or no sound at all.

Examples of a known-working buzzers.

- Hcm1205x Miniature Buzzer 5v (http://www.rapidonline.com/Audio-Visual/Hcm1205x-Miniature-Buzzer-5v-35-0055)
- 5V Electromagnetic Active Buzzer Continuous Beep (http://www.banggood.com/10Pcs-5V-Electromagnetic-Active-Buzzer-Continuous-Beep-Continuously-p-943524.html)
- Radio Shack Model: 273-074 PC-BOARD 12VDC (3-16v) 70DB PIEZO BUZZER (http://www.radioshack.com/pc-board-12vdc-70db-piezo-buzzer/2730074.html#.VIAtpzHF_Si)
- MultiComp MCKPX-G1205A-3700 TRANSDUCER, THRU-HOLE, 4V, 30MA (http://uk.farnell.com/multicomp/mckpx-g1205a-3700/transducer-thru-hole-4v-30ma/dp/2135914?CMP=i-bf9f-00001000)
- 3-24V Piezo Electronic Tone Buzzer Alarm 95DB (http://www.banggood.com/3-24V-Piezo-Electronic-Tone-Buzzer-Alarm-95DB-Continuous-Sound-p-919348.html)

Connections

Naze32

Connect a supported buzzer directly to the BUZZ pins. Observe polarity. Also if you are working with flight controller outside of a craft, on a bench for example, you need to supply 5 volts and ground to one of the ESC connections or the buzzer will not function.

CC3D

Buzzer support on the CC3D requires that a buzzer circuit be created to which the input is PA15.
PA15 is unused and not connected according to the CC3D Revision A schematic.
Connecting to PA15 requires careful soldering.

See the CC3D - buzzer circuit.pdf (Wiring/CC3D%20-%20buzzer%20circuit.pdf) for details.

Command Line Interface (CLI)

Cleanflight has a command line interface (CLI) that can be used to change settings and configure the FC.

Accessing the CLI.

The CLI can be accessed via the GUI tool or via a terminal emulator connected to the CLI serial port.

1. Connect your terminal emulator to the CLI serial port (which, by default, is the same as the MSP serial port)
2. Use the baudrate specified by msp_baudrate (115200 by default).
3. Send a # character.

To save your settings type in 'save', saving will reboot the flight controller.

To exit the CLI without saving power off the flight controller or type in 'exit'.

To see a list of other commands type in 'help' and press return.

To dump your configuration (including the current profile), use the 'dump' command.

See the other documentation sections for details of the cli commands and settings that are available.

Backup via CLI

Disconnect main power, connect to cli via USB/FTDI.

dump using cli

```
rateprofile 0
profile 0
dump
```

dump profiles using cli if you use them

```
profile 1
dump profile
profile 2
dump profile
```

dump rate profiles using cli if you use them

```
rateprofile 1
dump rates
rateprofile 2
dump rates
```

copy screen output to a file and save it.

Restore via CLI.

Use the cli `defaults` command first.

When restoring from a backup it is a good idea to do a dump of the latest defaults so you know what has changed - if you do this each time a firmware release is created youwill be able to see the cli changes between firmware versions. For instance, in December 2014 the default GPS navigation PIDs changed. If you blindly restore your backup you would not benefit from these new defaults.

Use the CLI and send all the output from the saved backup commands.

Do not send the file too fast, if you do the FC might not be able to keep up when using USART adapters (including built in ones) since there is no hardware serial flow control.

You may find you have to copy/paste a few lines at a time.

Repeat the backup process again!

Compare the two backups to make sure you are happy with your restored settings.

Re-apply any new defaults as desired.

CLI Command Reference

Click on a command to jump to the relevant documentation page.

Command	Description
`1wire <esc>`	passthrough 1wire to the specified esc
`adjrange` (Inflight%20Adjustments.md)	show/set adjustment ranges settings
`aux` (Modes.md)	show/set aux settings
`mmix` (Mixer.md)	design custom motor mixer
`smix` (Mixer.md)	design custom servo mixer
`color` (LedStrip.md)	configure colors
`defaults`	reset to defaults and reboot
`dump`	print configurable settings in a pastable form
`exit`	
`feature`	list or -val or val
`get`	get variable value
`gpspassthrough` (Gps.md)	passthrough gps to serial
`help`	
`led` (LedStrip.md)	configure leds
`map` (Rx.md)	mapping of rc channel order
`mixer` (Mixer.md)	mixer name or list
`mode_color` (LedStrip.md)	configure mode colors
`motor`	get/set motor output value
`play_sound` (Buzzer.md)	index, or none for next
`profile` (Profiles.md)	index (0 to 2)
`rateprofile` (Profiles.md)	index (0 to 2)
`rxrange` (Rx.md)	configure rx channel ranges (end-points)

rxfail (Rx.md)	show/set rx failsafe settings
save	save and reboot
serialpassthrough	serial passthrough mode, reset board to exit
set	name=value or blank or * for list
status	show system status
version	show version
serial (Serial.md)	configure serial ports
servo (Mixer.md)	configure servos
sd_info	sdcard info
tasks	show task stats

CLI Variable Reference

Click on a variable to jump to the relevant documentation page.

Variable	Description/Units	Min	Max	Defau
looptime	This is the main loop time (in us). Changing this affects PID effect with some PID controllers (see PID section for details). Default of 3500us/285Hz should work for everyone. Setting it to zero does not limit loop time, so it will go as fast as possible.	0	9000	3500
emf_avoidance	Default value is OFF for 72MHz processor speed. Setting this to ON increases the processor speed, to move the 6th harmonic away from 432MHz.	OFF	ON	OFF
i2c_highspeed	Enabling this feature speeds up IMU speed significantly and faster looptimes are possible.	OFF	ON	ON
gyro_sync (Pid%20tuning.md)	This option enables gyro_sync feature. In this case the loop will be synced to gyro refresh rate. Loop will always wait for the newest gyro measurement. Use gyro_lpf and gyro_sync_denom determine the gyro refresh rate. Note that different targets have different limits. Setting too high refresh rate can mean that FC cannot keep up with the gyro and higher gyro_sync_denom is needed.	OFF	ON	ON
	This is an important number to set in order to avoid trimming receiver/transmitter. Most standard receivers will have this			

mid_rc (Rx.md)	at 1500, however Futaba transmitters will need this set to 1520. A way to find out if this needs to be changed, is to clear all trim/subtrim on transmitter, and connect to GUI. Note the value most channels idle at - this should be the number to choose. Once midrc is set, use subtrim on transmitter to make sure all channels (except throttle of course) are centered at midrc value.	1200	1700	1500
min_check (Controls.md)	These are min/max values (in us) which, when a channel is smaller (min) or larger (max) than the value will activate various RC commands, such as arming, or stick configuration. Normally, every RC channel should be set so that min = 1000us, max = 2000us. On most transmitters this usually means 125% endpoints. Default check values are 100us above/below this value.	0	2000	1100
max_check (Controls.md)	These are min/max values (in us) which, when a channel is smaller (min) or larger (max) than the value will activate various RC commands, such as arming, or stick configuration. Normally, every RC channel should be set so that min = 1000us, max = 2000us. On most transmitters this usually means 125% endpoints. Default check values are 100us above/below this value.	0	2000	1900
rssi_channel (Rssi.md)	RX channel containing the RSSI signal	0	18	0
rssi_scale (Rssi.md)	When using ADC RSSI, the raw ADC value will be divided by rssi_scale in order to get the RSSI percentage. RSSI scale is therefore the ADC raw value for 100% RSSI.	1	255	30
rssi_invert (Rssi.md)	When using PWM RSSI or ADC RSSI, determines if the signal is inverted (Futaba, FrSKY)	OFF	ON	ON

rc_smoothing	Interpolation of Rc data during looptimes when there are no new updates. This gives smoother RC input to PID controller and cleaner PIDsum	OFF	ON	ON
rx_min_usec (Rx.md)	Defines the shortest pulse width value used when ensuring the channel value is valid. If the receiver gives a pulse value lower than this value then the channel will be marked as bad and will default to the value of mid_rc.	750	2250	885
rx_max_usec (Rx.md)	Defines the longest pulse width value used when ensuring the channel value is valid. If the receiver gives a pulse value higher than this value then the channel will be marked as bad and will default to the value of mid_rc.	750	2250	2115
serialrx_provider (Rx.md)	When feature SERIALRX is enabled, this allows connection to several receivers which output data via digital interface resembling serial. Possible values: SPEK1024, SPEK2048, SBUS, SUMD, XB-B, XB-B-RJ01, IBUS			SPEK1024
sbus_inversion (Rx.md)	Standard SBUS (Futaba, FrSKY) uses an inverted signal. Some OpenLRS receivers produce a non-inverted SBUS signal. This setting is to support this type of receivers (including modified FrSKY). This only works on supported hardware (mainly F3 based flight controllers).	OFF	ON	ON
spektrum_sat_bind (Spektrum%20bind.md)	0 = disabled. Used to bind the spektrum satellite to RX	0	10	0
input_filtering_mode (Rx.md)	Filter out noise from OpenLRS Telemetry RX	OFF	ON	ON
min_throttle (Controls.md)	These are min/max values (in us) that are sent to esc when armed. Defaults of 1150/1850 are OK for everyone, for use with AfroESC, they could be set to 1064/1864.	0	2000	1150
	These are min/max values (in us) that are sent to esc when			

max_throttle (Controls.md)	armed. Defaults of 1150/1850 are OK for everyone, for use with AfroESC, they could be set to 1064/1864. If you have brushed motors, the value should be set to 2000.	0	2000	1850
min_command (Controls.md)	This is the PWM value sent to ESCs when they are not armed. If ESCs beep slowly when powered up, try decreasing this value. It can also be used for calibrating all ESCs at once.	0	2000	1000
servo_center_pulse	Servo midpoint	0	2000	1500
motor_pwm_rate	Output frequency (in Hz) for motor pins. Defaults are 400Hz for motor. If setting above 500Hz, will switch to brushed (direct drive) motors mode. For example, setting to 8000 will use brushed mode at 8kHz switching frequency. Up to 32kHz is supported. Default is 16000 for boards with brushed motors. Note, that in brushed mode, minthrottle is offset to zero. For brushed mode, set max_throttle to 2000.	50	32000	400
servo_pwm_rate	Output frequency (in Hz) servo pins. Default is 50Hz. When using tricopters or gimbal with digital servo, this rate can be increased. Max of 498Hz (for 500Hz pwm period), and min of 50Hz. Most digital servos will support for example 330Hz.	50	498	50
3d_deadband_low	Low value of throttle deadband for 3D mode (when stick is in the 3d_deadband_throttle range, the fixed values of 3d_deadband_low / _high are used instead)	0	2000	1406
3d_deadband_high	High value of throttle deadband for 3D mode (when stick is in the deadband range, the value in 3d_neutral is used instead)	0	2000	1514
3d_neutral	Neutral (stop) throttle value for 3D mode	0	2000	1460
	Disabled by default, enabling (setting to 1) allows disarming by throttle low + roll. This could			

retarded_arm	be useful for mode-1 users and non-acro tricopters, where default arming by yaw could move tail servo too much.	OFF	ON	OFF
disarm_kill_switch	Enabled by default. Disarms the motors independently of throttle value. Setting to 0 reverts to the old behaviour of disarming only when the throttle is low. Only applies when arming and disarming with an AUX channel.	OFF	ON	ON
auto_disarm_delay	Delay before automatic disarming	0	60	5
max_arm_angle	Maximum horizontal angle before arming is disabled	0	180	25
small_angle	If the copter tilt angle exceed this value the copter will refuse to arm. default is 25°.	0	180	25
fixedwing_althold_dir	Used for fixed-wing aircrafts. Determines of the correction value applied to throttle in alititude hold mode should be inverted.	-1	1	1
reboot_character	Special character used to trigger reboot	48	126	82
gps_provider (Gps.md)	GPS standard. Possible values: NMEA, UBLOX			NMEA
gps_sbas_mode (Gps.md)	Ground assistance type. Possible values: AUTO, EGNOS, WAAS, MSAS, GAGAN			AUTO
gps_auto_config (Gps.md)	Enable automatic configuration of UBlox GPS receivers.	OFF	ON	ON
gps_auto_baud	Enable automatic detection of GPS baudrate.	OFF	ON	OFF
gps_pos_p	GPS Position hold: P parameter	0	200	15
gps_pos_i	GPS Position hold: I parameter	0	200	0
gps_pos_d	GPS Position hold: D parameter	0	200	0
gps_posr_p	GPS Position hold rate: P parameter	0	200	34
gps_posr_i	GPS Position hold rate: I parameter	0	200	14
gps_posr_d	GPS Position hold rate: D parameter	0	200	53
gps_nav_p	GPS Navigation: P parameter	0	200	25
gps_nav_i	GPS Navigation: I parameter	0	200	33
gps_nav_d	GPS Navigation: D parameter	0	200	83

gps_wp_radius	GPS Navigation: waypoint radius	0	2000	200
nav_controls_heading	GPS Navigation: should the craft's heading follow the flying direction.	OFF	ON	ON
nav_speed_min	GPS Navigation: minimum moving speed	10	2000	100
nav_speed_max	GPS Navigation: maximum moving speed	10	2000	300
nav_slew_rate	GPS Navigation: maximum angle correction value. Lower slew rate stops the craft from rotating too quickly.	0	100	30
telemetry_switch	When an AUX channel is used to change serial output & baud rate (MSP / Telemetry). OFF: Telemetry is activated when armed. ON: Telemetry is activated by the AUX channel.	OFF	ON	OFF
ibus_report_cell_voltage (Telemetry.md)	Determines if the voltage reported is Vbatt or calculated average cell voltage (Flysky ibus telemtery)	OFF	ON	OFF
telemetry_inversion (Telemetry.md)	Determines if the telemetry signal is inverted (Futaba, FrSKY)	OFF	ON	OFF
telemetry_send_cells	Emulates FrSky FLVSS individual cell voltages telemetry	OFF	ON	OFF
frsky_default_lattitude	OpenTX needs a valid set of coordinates to show compass value. A fake value defined in this setting is sent while no fix is acquired.	-90	90	0
frsky_default_longitude	OpenTX needs a valid set of coordinates to show compass value. A fake value defined in this setting is sent while no fix is acquired.	-180	180	0
frsky_coordinates_format	FRSKY_FORMAT_DMS (default), FRSKY_FORMAT_NMEA			FRSKY_FORM
frsky_unit	IMPERIAL (default), METRIC			IMPERIAL
frsky_vfas_precision (Telemetry.md)	Set to 1 to send raw VBat value in 0.1V resolution for receivers that can handle it, or 0 (default) to use the standard method	0	1	0
hott_alarm_sound_interval	Battery alarm delay in seconds for Hott telemetry	0	120	5
	Battery capacity in mAH. This value is used in conjunction			

battery_capacity (Battery.md)	with the current meter to determine remaining battery capacity.	0	20000	0
vbat_scale (Battery.md)	Result is Vbatt in 0.1V steps. 3.3V = ADC Vref, 4095 = 12bit adc, 110 = 11:1 voltage divider (10k:1k) x 10 for 0.1V. Adjust this slightly if reported pack voltage is different from multimeter reading. You can get current voltage by typing "status" in cli.	0	255	110
vbat_max_cell_voltage (Battery.md)	Maximum voltage per cell, used for auto-detecting battery voltage in 0.1V units, default is 43 (4.3V)	10	50	43
vbat_min_cell_voltage (Battery.md)	Minimum voltage per cell, this triggers battery-critical alarms, in 0.1V units, default is 33 (3.3V)	10	50	33
vbat_warning_cell_voltage (Battery.md)	Warning voltage per cell, this triggers battery-warning alarms, in 0.1V units, default is 35 (3.5V)	10	50	35
vbat_hysteresis (Battery.md)	Sets the hysteresis value for low-battery alarms, in 0.1V units, default is 1 (0.1V)	10	250	1
current_meter_scale (Battery.md)	This sets the output voltage to current scaling for the current sensor in 0.1 mV/A steps. 400 is 40mV/A such as the ACS756 sensor outputs. 183 is the setting for the uberdistro with a 0.25mOhm shunt.	-10000	10000	400
current_meter_offset (Battery.md)	This sets the output offset voltage of the current sensor in millivolts.	0	3300	0
multiwii_current_meter_output (Battery.md)	Default current output via MSP is in 0.01A steps. Setting this to 1 causes output in default multiwii scaling (1mA steps).	OFF	ON	OFF
current_meter_type (Battery.md)	ADC (default), VIRTUAL, NONE. The virtual current sensor, once calibrated, estimates the current value from throttle position. When running on non-default hardware or adding support for new sensors/sensor boards, these values are used for			ADC

align_gyro	sensor orientation. When carefully understood, these values can also be used to rotate (in 90deg steps) or flip the board. Possible values are: DEFAULT, CW0, CW90, CW180, CW270, CW0FLIP, CW90FLIP, CW180FLIP, CW270FLIP.			DEFAULT
align_acc	When running on non-default hardware or adding support for new sensors/sensor boards, these values are used for sensor orientation. When carefully understood, these values can also be used to rotate (in 90deg steps) or flip the board. Possible values are: DEFAULT, CW0, CW90, CW180, CW270, CW0FLIP, CW90FLIP, CW180FLIP, CW270FLIP.			DEFAULT
align_mag	When running on non-default hardware or adding support for new sensors/sensor boards, these values are used for sensor orientation. When carefully understood, these values can also be used to rotate (in 90deg steps) or flip the board. Possible values are: DEFAULT, CW0, CW90, CW180, CW270, CW0FLIP, CW90FLIP, CW180FLIP, CW270FLIP.			DEFAULT
align_board_roll	Arbitrary board rotation in degrees, to allow mounting it sideways / upside down / rotated etc	-180	360	0
align_board_pitch	Arbitrary board rotation in degrees, to allow mounting it sideways / upside down / rotated etc	-180	360	0
align_board_yaw	Arbitrary board rotation in degrees, to allow mounting it sideways / upside down / rotated etc	-180	360	0
max_angle_inclination	This setting controls max inclination (tilt) allowed in angle (level) mode. default 500 (50 degrees).	100	900	500
	Hardware lowpass filter cutoff frequency for gyro. Allowed			

gyro_lpf (PID%20tuning.md)	values depend on the driver - For example MPU6050 allows 10HZ,20HZ,42HZ,98HZ,188HZ. If you have to set gyro lpf below 42Hz generally means the frame is vibrating too much, and that should be fixed first.	10HZ	188HZ	42HZ
gyro_soft_lpf	Software lowpass filter cutoff frequency for gyro. Default is 60Hz. Set to 0 to disable.	0	500	60
moron_threshold	When powering up, gyro bias is calculated. If the model is shaking/moving during this initial calibration, offsets are calculated incorrectly, and could lead to poor flying performance. This threshold (default of 32) means how much average gyro reading could differ before re-calibration is triggered.	0	128	32
imu_dcm_kp	Inertial Measurement Unit KP Gain	0	20000	2500
imu_dcm_ki	Inertial Measurement Unit KI Gain	0	20000	0
alt_hold_deadband	Altitude will be held when throttle is centered with an error margin defined in this parameter.	1	250	40
alt_hold_fast_change	Authorise fast altitude changes. Should be disabled when slow changes are prefered, for example for aerial photography.	OFF	ON	ON
deadband (Controls.md)	These are values (in us) by how much RC input can be different before it's considered valid for roll and pitch axis. For transmitters with jitter on outputs, this value can be increased. Defaults are zero, but can be increased up to 10 or so if rc inputs twitch while idle. This value is applied either side of the centrepoint.	0	32	0
yaw_deadband (Controls.md)	These are values (in us) by how much RC input can be different before it's considered valid for the yaw axis. For transmitters with jitter on outputs, this value can be increased. Defaults are zero, but can be increased up to	0	100	0

	10 or so if rc inputs twitch while idle. This value is applied either side of the centrepoint.			
yaw_control_direction	Use if you need to inverse yaw control direction.	-1	1	1
3d_deadband_throttle	Throttle signal will be held to a fixed value when throttle is centered with an error margin defined in this parameter.	0	2000	50
throttle_correction_value	The throttle_correction_value will be added to the throttle input. It will be maximal at the throttle_correction_angle and over, null when the copter is leveled and proportional in bewteen. The angle is set with 0.1 deg steps from 1 to 900, ie : 300 = 30.0 deg, 225 = 22.5 deg.	0	150	0
throttle_correction_angle	The throttle_correction_value will be added to the throttle input. It will be maximal at the throttle_correction_angle and over, null when the copter is leveled and proportional in bewteen. The angle is set with 0.1 deg steps from 1 to 900, ie : 300 = 30.0 deg, 225 = 22.5 deg.	1	900	800
pid_at_min_throttle	If enabled, the copter will process the pid algorithm at minimum throttle. Cannot be used when retarded_arm is enabled.	OFF	ON	ON
yaw_motor_direction	Use if you need to inverse yaw motor direction.	-1	1	1
yaw_jump_prevention_limit	Prevent yaw jumps during yaw stops and rapid YAW input. To disable set to 500. Adjust this if your aircraft 'skids out'. Higher values increases YAW authority but can cause roll/pitch instability in case of underpowered UAVs. Lower values makes yaw adjustments more gentle but can cause UAV unable to keep heading	80	500	200
tri_unarmed_servo (Controls.md)	On tricopter mix only, if this is set to 1, servo will always be correcting regardless of armed	OFF	ON	ON

	state. to disable this, set it to 0.			
servo_lowpass_freq (Mixer.md)	Selects the servo PWM output cutoff frequency. Valid values range from 10 to 400. This is a fraction of the loop frequency in 1/1000ths. For example, 40 means 0.040. The cutoff frequency can be determined by the following formula: Frequency = 1000 * servo_lowpass_freq / looptime	10	400	400
servo_lowpass_enable (Mixer.md)	Disabled by default.	OFF	ON	OFF
default_rate_profile (Profiles.md)	Default = profile number	0	2	0
rc_rate (Profiles.md)	Rate value for all RC directions	0	250	90
rc_expo (Profiles.md)	Exposition value for all RC directions	0	100	65
rc_yaw_expo	Yaw exposition value	0	100	0
thr_mid (Profiles.md)	Throttle value when the stick is set to mid-position. Used in the throttle curve calculation.	0	100	50
thr_expo (Profiles.md)	Throttle exposition value	0	100	0
roll_rate	Roll rate value		100	40
pitch_rate (Profiles.md)	Pitch rate value		100	40
yaw_rate (Profiles.md)	Yaw rate value	0	255	0
tpa_rate (Profiles.md)	Throttle PID attenuation reduces influence of P on ROLL and PITCH as throttle increases. For every 1% throttle after the TPA breakpoint, P is reduced by the TPA rate.	0	100	0
tpa_breakpoint (Profiles.md)	See tpa_rate.	1000	2000	1500
failsafe_delay (Failsafe.md)	Time in deciseconds to wait before activating failsafe when signal is lost. See Failsafe documentation (Failsafe.md#failsafe_delay).	0	200	10
failsafe_off_delay (Failsafe.md)	Time in deciseconds to wait before turning off motors when failsafe is activated. See Failsafe	0	200	200

	documentation (Failsafe.md#failsafe_off_delay).			
failsafe_throttle (Failsafe.md)	Throttle level used for landing when failsafe is enabled. See Failsafe documentation (Failsafe.md#failsafe_throttle).	1000	2000	1000
failsafe_kill_switch (Failsafe.md)	Set to ON to use an AUX channel as a faisafe kill switch.	OFF	ON	OFF
failsafe_throttle_low_delay (Failsafe.md)	Activate failsafe when throttle is low and no RX data has been received since this value, in 10th of seconds	0	300	100
failsafe_procedure (Failsafe.md)	0 = Autolanding (default). 1 = Drop.	0	1	0
gimbal_mode	When feature SERVO_TILT is enabled, this can be either NORMAL or MIXTILT			NORMAL
acc_hardware	This is used to suggest which accelerometer driver should load, or to force no accelerometer in case gyro-only flight is needed. Default (0) will attempt to auto-detect among enabled drivers. Otherwise, to force a particular device, set it to 2 for ADXL345, 3 for MPU6050 integrated accelerometer, 4 for MMA8452, 5 for BMA280, 6 for LSM303DLHC, 7 for MPU6000, 8 for MPU6500 or 1 to disable accelerometer alltogether - resulting in gyro-only operation.	0	9	0
acc_cut_hz	Set the Low Pass Filter factor for ACC. Reducing this value would reduce ACC noise (visible in GUI), but would increase ACC lag time. Zero = no filter	0	200	15
accxy_deadband	Deadband applied to accelerometrer measurements to reduce integration drift and vibration influence	0	100	40
accz_deadband	Deadband applied to accelerometrer measurements to reduce integration drift and vibration influence	0	100	40
accz_lpf_cutoff	Cutoff frequency used in the low-pass filtering of accelerometer measurements.	1	20	5
	Determines the method used to			

acc_unarmedcal	calculate gravitational compensation on the Z axis in the Inertial Measurement Unit's acceleration calculation. The method used when set to ON takes account of the armed status.	OFF	ON	ON
acc_trim_pitch	Accelerometer trim (Pitch)	-300	300	0
acc_trim_roll	Accelerometer trim (Roll)	-300	300	0
baro_tab_size	Pressure sensor sample count.	0	48	21
baro_noise_lpf	barometer low-pass filter cut-off frequency in Hz. Ranges from 0 to 1 ; default 0.6	0	1	0.6
baro_cf_vel	Velocity sensor mix in altitude hold. Determines the influence accelerometer and barometer sensors have in the velocity estimation. Values from 0 to 1; 1 for pure accelerometer altitude, 0 for pure barometer altitude.	0	1	0.985
baro_cf_alt	Altitude sensor mix in altitude hold. Determines the influence accelerometer and barometer sensors have in the altitude estimation. Values from 0 to 1; 1 for pure accelerometer altitude, 0 for pure barometer altitude.	0	1	0.965
baro_hardware	0 = Default, use whatever mag hardware is defined for your board type ; 1 = None, 2 = BMP085, 3 = MS5611, 4 = BMP280	0	4	0
mag_hardware	0 = Default, use whatever mag hardware is defined for your board type ; 1 = None, disable mag ; 2 = HMC5883 ; 3 = AK8975 ; 4 = AK8963 (for versions <= 1.7.1: 1 = HMC5883 ; 2 = AK8975 ; 3 = None, disable mag)	0	4	0
mag_declination	Current location magnetic declination in dddmm format. For example, -6deg 37min = -637 for Japan. Leading zeros not required. Get your local magnetic declination here: http://magnetic-declination.com/	-18000	18000	0

p_pitch (PID%20tuning.md)	Pitch P parameter	0	200	40
i_pitch (PID%20tuning.md)	Pitch I parameter	0	200	30
d_pitch (PID%20tuning.md)	Pitch D parameter	0	200	23
p_roll (PID%20tuning.md)	Roll P parameter	0	200	40
i_roll (PID%20tuning.md)	Roll I parameter	0	200	30
d_roll (PID%20tuning.md)	Roll D parameter	0	200	23
p_yaw (PID%20tuning.md)	Yaw P parameter	0	200	85
i_yaw (PID%20tuning.md)	Yaw I parameter	0	200	45
d_yaw (PID%20tuning.md)	Yaw D parameter	0	200	0
p_alt (PID%20tuning.md)	Altitude P parameter (Baro / Sonar altitude hold)	0	200	50
i_alt (PID%20tuning.md)	Altitude I parameter (Baro / Sonar altitude hold)	0	200	0
d_alt (PID%20tuning.md)	Altitude D parameter (Baro / Sonar altitude hold)	0	200	0
p_level (PID%20tuning.md)	Level P parameter (Angle / horizon modes)	0	200	20
i_level (PID%20tuning.md)	Level I parameter (Angle / horizon modes)	0	200	10
d_level (PID%20tuning.md)	Level D parameter (Angle / horizon modes)	0	200	100
p_vel (PID%20tuning.md)	Velocity P parameter (Baro / Sonar altitude hold)	0	200	120
i_vel (PID%20tuning.md)	Velocity I parameter (Baro / Sonar altitude hold)	0	200	45
d_vel (PID%20tuning.md)	Velocity D parameter (Baro / Sonar altitude hold)	0	200	1
yaw_p_limit	Limiter for yaw P term. This parameter is only affecting PID controller MW23. To disable set to 500 (actual default).	100	500	500
dterm_cut_hz (PID%20tuning.md)	Lowpass cutoff filter for Dterm for all PID controllers	0	500	0
gtune_loP_rll (Gtune.md)	GTune: Low Roll P limit	10	200	10
gtune_loP_ptch (Gtune.md)	GTune: Low Pitch P limit	10	200	10
gtune_loP_yw (Gtune.md)	GTune: Low Yaw P limit	10	200	10
gtune_hiP_rll (Gtune.md)	GTune: High Roll P limit	0	200	100
gtune_hiP_ptch (Gtune.md)	GTune: High Pitch P limit	0	200	100
gtune_hiP_yw (Gtune.md)	GTune: High Yaw P limit	0	200	100
gtune_pwr (Gtune.md)	Strength of each Gtune adjustment	0	10	0
gtune_settle_time (Gtune.md)	GTune settling time in milliseconds	200		450
gtune_average_cycles (Gtune.md)	Looptime cycles for gyro average calculation. Default = 16.	8	128	16
	Blackbox logging rate			

blackbox_rate_num (Blackbox.md)	numerator. Use num/denom settings to decide if a frame should be logged, allowing control of the portion of logged loop iterations	1	32	1
blackbox_rate_denom (Blackbox.md)	Blackbox logging rate denominator. See blackbox_rate_num.	1	32	1
blackbox_device (Blackbox.md)	SERIAL, SPIFLASH, SDCARD (default)			SDCARD
magzero_x	Magnetometer calibration X offset	-32768	32767	0
magzero_y	Magnetometer calibration Y offset	-32768	32767	0
magzero_z	Magnetometer calibration Z offset	-32768	32767	0

Configuration

Cleanflight is configured primarily using the Cleanflight Configurator GUI.

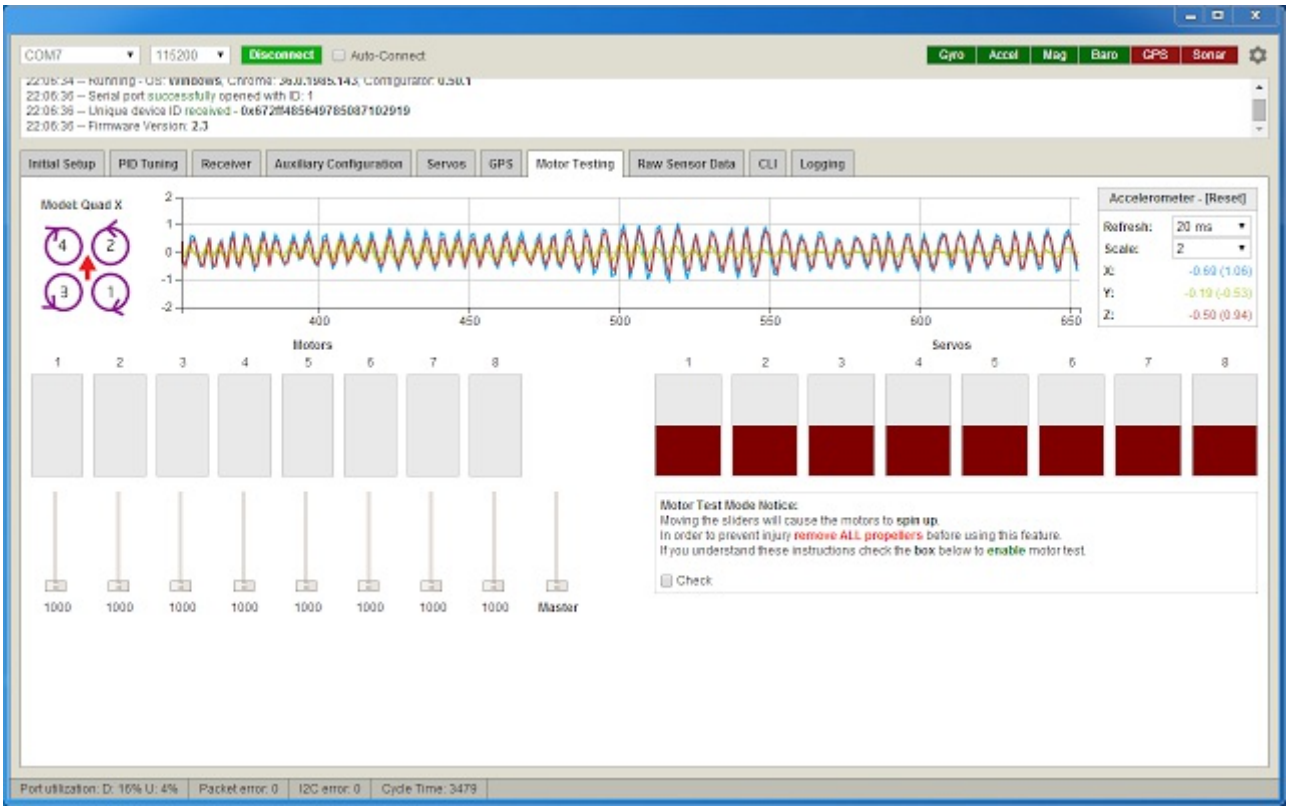
Both the command line interface and GUI are accessible by connecting to a serial port on the target,
be it a USB virtual serial port, physical hardware UART port or a SoftSerial port.

See the Serial section for more information and see the Board specific sections for details of the serial ports available on the board you are using.

The GUI cannot currently configure all aspects of the system, the CLI must be used to enable or configure
some features and settings.

Due to ongoing development, the fact that the GUI cannot yet backup all your settings and automatic chrome updates of the GUI app it is highly advisable to backup your settings (using the CLI) so that when a new version of the configurator or firmware is released you can re-apply your settings.

GUI



The GUI tool is the preferred way of configuration. The GUI tool also includes a terminal which can be used to interact with the CLI.

[Cleanflight Configurator on Chrome store (https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb)]

If you cannot use the latest version of the GUI to access the FC due to firmware compatibility issues you can still access the FC via the CLI to backup your settings, or you can install an old version of the configurator.

Old versions of the configurator can be downloaded from the configurator releases page:
https://github.com/cleanflight/cleanflight-configurator/releases
See the README file that comes with the configurator for installation instructions.

CLI

Cleanflight can also be configured by a command line interface.

See the CLI section (Cli.md) of the documentation for more details.

Controls

Arming

When armed, the aircraft is ready to fly and the motors will spin when throttle is applied. The motors will
spin at a slow speed when armed (this feature may be disabled by setting MOTOR_STOP, but for safety reasons,
that is not recommended).

By default, arming and disarming is done using stick positions. (NOTE: this feature is disabled when using a
switch to arm.)

Some conditions will disable arming. In this case the Warning LED on the board will flash a certain number of times, indicating what the condition is:

Reason for disabled Arming	LED Flashes
CLI is active in the configurator	2
Failsafe mode is active	3
The aircraft has landed in failsafe mode	3
Maximum arming angle is exceeded	4
Calibration is active	5
The system is overloaded	6

Stick Positions

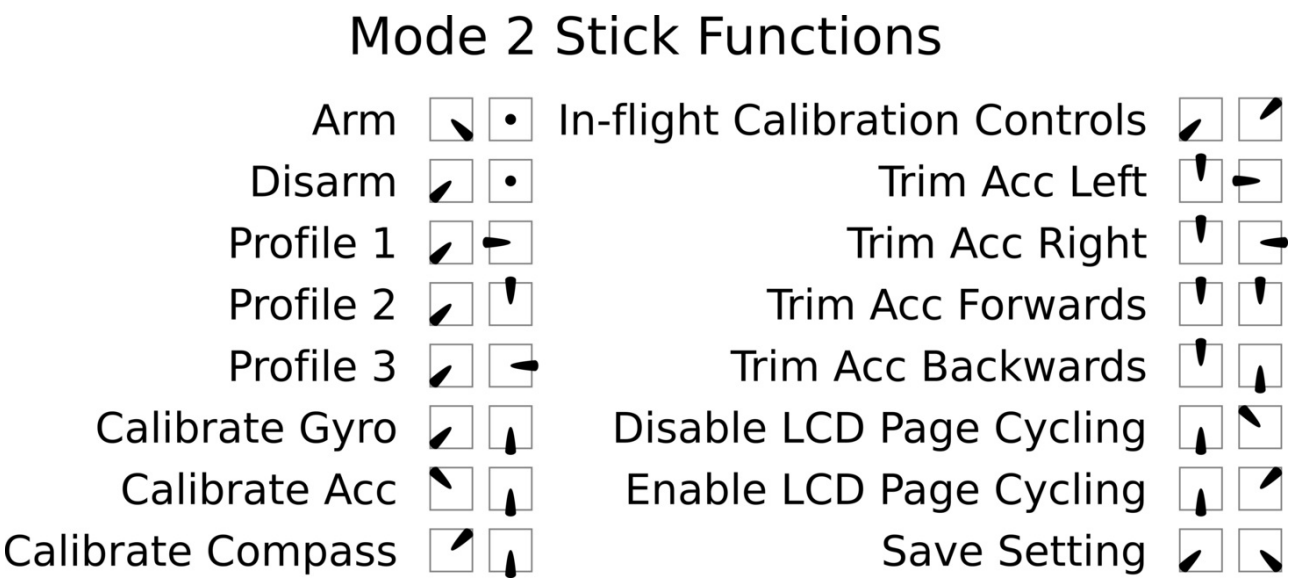
The three stick positions are:

Position	Approx. Channel Input
LOW	1000
CENTER	1500
HIGH	2000

The stick positions are combined to activate different functions:

Function	Throttle	Yaw	Pitch	Roll
ARM	LOW	HIGH	CENTER	CENTER
DISARM	LOW	LOW	CENTER	CENTER
Profile 1	LOW	LOW	CENTER	LOW
Profile 2	LOW	LOW	HIGH	CENTER
Profile 3	LOW	LOW	CENTER	HIGH
Calibrate Gyro	LOW	LOW	LOW	CENTER
Calibrate Acc	HIGH	LOW	LOW	CENTER
Calibrate Mag/Compass	HIGH	HIGH	LOW	CENTER
Inflight calibration controls	LOW	LOW	HIGH	HIGH
Trim Acc Left	HIGH	CENTER	CENTER	LOW

Trim Acc Right	HIGH	CENTER	CENTER	HIGH
Trim Acc Forwards	HIGH	CENTER	HIGH	CENTER
Trim Acc Backwards	HIGH	CENTER	LOW	CENTER
Disable LCD Page Cycling	LOW	CENTER	HIGH	LOW
Enable LCD Page Cycling	LOW	CENTER	HIGH	HIGH
Save setting	LOW	LOW	LOW	HIGH



History

Intial stick commands, came from MultiWII but the original code no-longer has direct links.

The original documents as listed below can be found here
https://code.google.com/archive/p/multiwii/source/default/source

- `svn/branches/Hamburger/MultiWii-StickConfiguration-23_v0-5772156649.pdf`
- `multiwii/branches/Hamburger/MultiWii-StickConfiguration-23_v0-5772156649.odp`

Yaw control

While arming/disarming with sticks, your yaw stick will be moving to extreme values. In order to prevent your craft
from trying to yaw during arming/disarming while on the ground, your yaw input will not cause the craft to yaw when the
throttle is LOW (i.e. below the `min_check` setting).

For tricopters, you may want to retain the ability to yaw while on the ground, so that you can verify that your tail
servo is working correctly before takeoff. You can do this by setting `tri_unarmed_servo` to `ON` on the CLI (this is the
default). If you are having issues with your tail rotor contacting the ground during arm/disarm, you can set this to
`0` instead. Check this table to decide which setting will suit you:

<table>
<tr>
<th colspan="5">Is yaw control of the tricopter allowed?</th>
</tr>
<tr>
<th></th><th colspan="2">Disarmed</th><th colspan="2">Armed</th>
</tr>
<tr>
<th></th><th>Throttle low</th><th>Throttle normal</th><th>Throttle low</th>
<th>Throttle normal</th>
</tr>
<tr>
<td rowspan="2">tri_unarmed_servo = OFF</td><td>No</td><td>No</td><td>No</td><td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>No</td><td>No</td><td>No</td><td>Yes</td>
</tr>
<tr>
<td rowspan="2">tri_unarmed_servo = ON</td><td>Yes</td><td>Yes</td><td>Yes</td><td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Yes</td><td>Yes</td><td>Yes</td><td>Yes</td>
</tr>
</table>

Throttle settings and their interaction

`min_command` -
With motor stop enabled this is the command sent to the esc's when the throttle is below min_check or disarmed. With motor stop disabled, this is the command sent only when the copter is disarmed. This must be set well below motors spinning for safety.

`min_check` -
With switch arming mode is in use, lowering your throttle below min_check will result in motors spinning at min_throttle. When using the default stick arming, lowering your throttle below min_check will result in motors spinning at min_throttle and yaw being disabled so that you may arm/disarm. With motor stop enabled, lowering your throttle below min_check will also result in motors off and the esc's being sent min_command. Min_check must be set to a level that is 100% reliably met by the throttle throw. A setting too low may result in a dangerous condition where the copter can’t be disarmed. It is ok to set this below min_throttle because the FC will automaticly scale the output to the esc's

`min_throttle` -
Typically set to just above reliable spin up of all motors. Sometimes this is set slightly higher for prop stall prevention during advanced maneuvers or sometimes considerably higher to produce a desired result. When armed with motor stop off, your motors will spin at this command so keep that in mind from a safety stand point.

`max_check` -
Throttle positions above this level will send max_command to the esc's.

`max_throttle` -
This is the max command to the esc's from the flight controller.

In depth videos explaining these terms are available from Joshua Bardwell here:

https://www.youtube.com/watch?v=WFU3VewGbbA

https://www.youtube.com/watch?v=YNRl0OTKRGA

Deadband

If yaw, roll or pitch sticks do not reliably return to centre or the radio has a lot of jitter around the centrepoint, deadband can be applied. The whole deadband value is applied *either side* of the center point rather than half the value above and half the value below. The deadband value will have an effect on stick endpoint values as the axis value will be reduced by the amount of deadband applied.

deadband -
Applied to roll, pitch.

`yaw_deadband`
Only applied to yaw.

Display

Cleanflight supports displays to provide information to you about your aircraft and cleanflight state.

When the aircraft is armed the display does not update so flight is not affected. When disarmed the display cycles between various pages.

There is currently no way to change the information on the pages, the list of pages or the time between pages - Code submissions via pull-requests are welcomed!

Supported Hardware

At this time no other displays are supported other than the SSD1306 / UG-2864HSWEG01.

Configuration

From the CLI enable the `DISPLAY` feature

```
feature DISPLAY
```

SSD1306 OLED displays

The SSD1306 display is a 128x64 OLED display that is visible in full sunlight, small and consumes very little current.
This makes it ideal for aircraft use.

There are various models of SSD1306 boards out there, they are not all equal and some require addtional modifications
before they work. Choose wisely!

Links to displays:

- banggood.com (http://www.banggood.com/0_96-Inch-4Pin-White-IIC-I2C-OLED-Display-Module-12864-LED-For-Arduino-p-958196.html) 0.96 Inch 4Pin White IIC I2C OLED Display Module 12864 LED For Arduino
- banggood.com (http://www.banggood.com/0_96-Inch-4Pin-IIC-I2C-Blue-OLED-Display-Module-For-Arduino-p-969147.html) 0.96 Inch 4Pin IIC I2C Blue OLED Display Module For Arduino
- wide.hk (http://www.wide.hk/products.php?product=I2C-0.96%22-OLED-display-module-%28-compatible-Arduino-%29) I2C 0.96" OLED display module
- witespyquad.gostorego.com (http://witespyquad.gostorego.com/accessories/readytofly-1-oled-128x64-pid-tuning-display-i2c.html) ReadyToFlyQuads 1" OLED Display
- multiwiicopter.com (http://www.multiwiicopter.com/products/1-oled) PARIS 1" OLED 128x64 PID tuning screen AIR

The banggood.com display is the cheapest at the time fo writing and will correctly send I2C ACK signals.

Crius CO-16

This display is best avoided but will work if you modify it.

Step 1

As supplied the I2C ack signal is not sent because the manufacturer did not bridge D1 and D2 together. To fix this solder
the two pins together as they enter the screen. Failure to do this will result is a screen that doesn't display anything.

Step 2

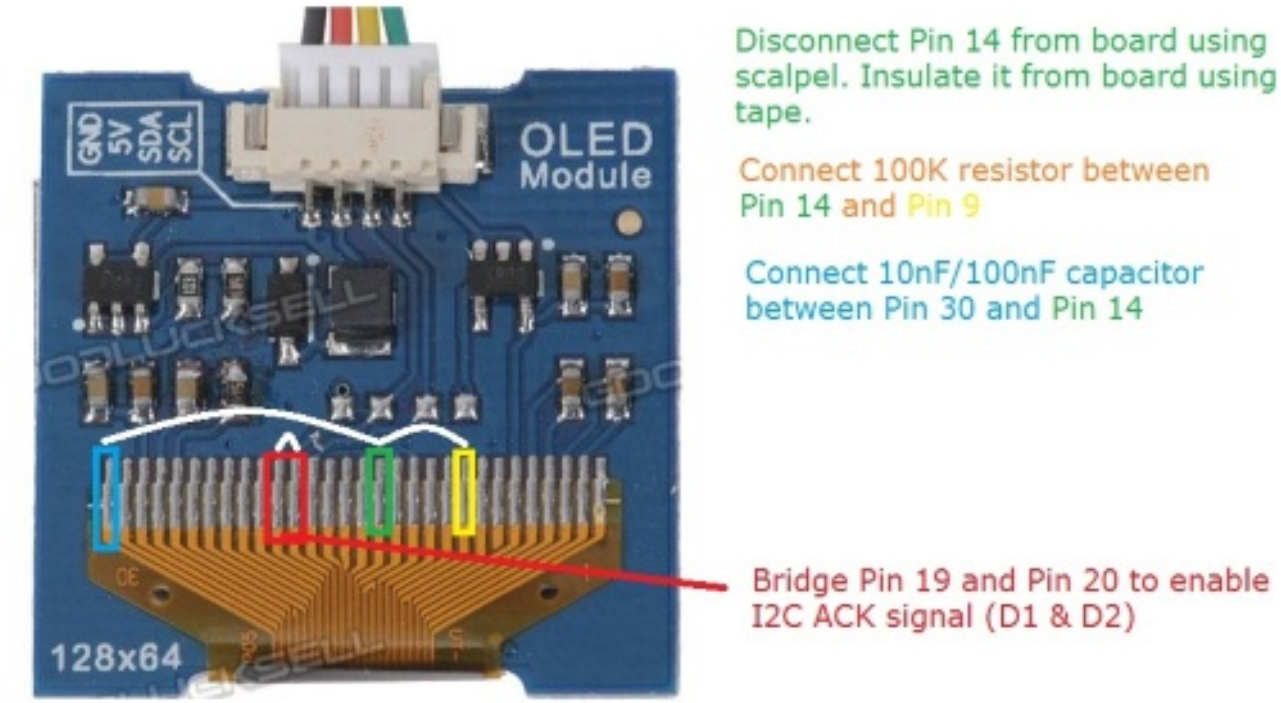
Pin 14 must be disconnected from the main board using a scalpel. Then connect a 10nF or 100nF capacitor between pins 30 and the
lifted pin 14.

Step 3

Connect a 100K resistor between Pin 9 and the lifted Pin 14.

Failure to perform steps 2 and 3 will result in a display that only works on power up some of the time any may display random dots
or other display corruption.

More can be read about this procedure here: http://www.multiwii.com/forum/viewtopic.php?f=6&t=2705&start=10



Connections

Connect +5v, Ground, I2C SDA and I2C SCL from the flight controller to the display.

On Naze32 rev 5 boards the SDA and SCL pads are underneath the board.

Failsafe

There are two types of failsafe:

1. Receiver based failsafe
2. Flight controller based failsafe

Receiver based failsafe is where you, from your transmitter and receiver, configure channels to output desired signals if your receiver detects signal loss and goes to the **failsafe mode**. The idea is that you set throttle and other controls so the aircraft descends in a controlled manner. See your receiver's documentation for this method.

Flight controller based failsafe is where the flight controller attempts to detect signal loss and/or the **failsafe mode** of your receiver and upon detection goes to **failsafe stage 1**. The idea is that the flight controller starts using **fallback settings** for all controls, which are set by you, using the CLI command `rxfail` (see rxfail (Rx.md#rx-loss-configuration) section in rx documentation) or the cleanflight-configurator GUI.

It is possible to use both types at the same time, which may be desirable. Flight controller failsafe can even help if your receiver signal wires come loose, get damaged or your receiver malfunctions in a way the receiver itself cannot detect.

Alternatively you may configure a transmitter switch to activate failsafe mode. This is useful for fieldtesting the failsafe system and as a ***PANIC*** switch when you lose orientation.

Flight controller failsafe system

This system has two stages.

Stage 1 is entered when **a flightchannel** has an ***invalid pulse length***, the receiver reports ***failsafe mode*** or there is ***no signal*** from the receiver. Fallback settings are applied to ***all channels*** and a short amount of time is provided to allow for recovery.

Note: Prior to entering **stage 1**, fallback settings are also applied to ***individual AUX channels*** that have invalid pulses.

Stage 1 can also directly be activated when a transmitter switch that is configured to control the failsafe mode is switched ON and `failsafe_switch_mode` is set to STAGE1. Stage 1 will be aborted if the switch is moved to the OFF position before Stage 2 is engaged (see next).

Stage 2 is entered when your craft is **armed** and **stage 1** persists longer then the configured guard time (`failsafe_delay`). All channels will remain at the applied fallback setting unless overruled by the chosen stage 2 procedure (`failsafe_procedure`).

Stage 2 is not activated until 5 seconds after the flight controller boots up. This is to prevent unwanted activation, as in the case of TX/RX gear with long bind procedures, before the RX sends out valid data.

Stage 2 can also directly be activated when a transmitter switch that is configured to control the failsafe mode is switched ON and `failsafe_switch_mode` is set to STAGE2.

Stage 2 will be aborted when it was due to:

- a lost RC signal and the RC signal has recovered.
- a transmitter failsafe switch was set to ON position and the switch is set to OFF position (and `failsafe_switch_mode` is *not* set to `KILL`).

Note that:

- At the end of the stage 2 procedure, the flight controller will be disarmed and re-arming will be locked until the signal from the receiver is restored for specific amount of time depending on the procedure (see below) AND the arming switch is in the OFF position (when an arm switch is in use).

- Prior to starting a stage 2 intervention it is checked if the throttle position was below `min_throttle` level for the last `failsafe_throttle_low_delay` seconds. If it was, the craft is assumed to be on the ground and is only disarmed. It may be re-armed without a power cycle.

Some notes about **SAFETY**:

- The failsafe system will be activated regardless of current throttle position. So when the failsafe intervention is aborted (RC signal restored/failsafe switch set to OFF) the current stick position will direct the craft !
- The craft may already be on the ground with motors stopped and that motors and props could spin again - the software does not currently detect if the craft is on the ground. Take care when using `MOTOR_STOP` feature. **Props will spin up without warning**, when armed with `MOTOR_STOP` feature ON (props are not spinning) ***and*** failsafe is activated !

Configuration

When configuring the flight controller failsafe, use the following steps:

1. Configure your receiver to do one of the following:

- Upon signal loss, send no signal/pulses over the channels
- Send an invalid signal over the channels (for example, send values lower than `rx_min_usec`)

and

- Ensure your receiver does not send out channel data that would cause a disarm by switch or sticks to be registered by the FC. This is especially important for those using a switch to arm.

See your receiver's documentation for direction on how to accomplish one of these.

- Configure one of the transmitter switches to activate the failsafe mode.

2. Set `failsafe_off_delay` to an appropriate value based on how high you fly

3. Set `failsafe_throttle` to a value that allows the aircraft to descend at approximately one meter per second (default is 1000 which should be throttle off).

These are the basic steps for flight controller failsafe configuration; see Failsafe Settings below for additional settings that may be changed.

Failsafe Settings

Failsafe delays are configured in 0.1 second steps.

1 step = 0.1sec

1 second = 10 steps

`failsafe_delay`

Guard time for failsafe activation after signal lost. This is the amount of time the flight controller waits to see if it begins receiving a valid signal again before activating failsafe.

`failsafe_off_delay`

Delay after failsafe activates before motors finally turn off. This is the amount of time 'failsafe_throttle' is active. If you fly at higher altitudes you may need more time to descend safely.

`failsafe_throttle`

Throttle level used for landing. Specify a value that causes the aircraft to descend at about 1M/sec. Default is set to 1000 which should correspond to throttle off.

`failsafe_switch_mode`

Configure the RC switched failsafe action. It can be one of:

- `STAGE1` - activates Stage 1 failsafe. RC controls are applied as configured for Stage 1 and the `failsafe_delay` guard time will have to elapse before Stage 2 is activated. This is useful if you want to simulate with a switch the exact signal loss failsafe behavior.
- `STAGE2` - skips Stage 1 and activates the Stage 2 procedure immediately (see `failsafe_procedure`). Useful if you want to assign instant auto-landing to a switch.
- `KILL` - disarms instantly (your craft will crash). Re-arming is locked for 1 second AND until the arming switch (if used) is moved to the OFF position. Similar effect can be achieved by:
    - setting `failsafe_switch_mode` to `STAGE2` and `failsafe_procedure` to `DROP`. The difference is that `DROP` locks re-arming for 3 seconds instead of 1.
    - setting `failsafe_switch_mode` to `STAGE2`, `failsafe_procedure` to `AUTO-LAND`, setting `failsafe_throttle` to 1000 and `failsafe_off_delay` to 0 (basically initiates an auto-landing but cuts it short immediately). This is not preferred method, since the reaction is slower and re-arming will be locked for 30 seconds.
    - using arm switch. This does not introduce re-arming locking.

`failsafe_throttle_low_delay`

Time throttle level must have been below 'min_throttle' to *only disarm* instead of *full failsafe procedure*.

Use standard RX usec values. See Rx documentation (Rx.md).

`failsafe_procedure`

- `DROP`: Just kill the motors and disarm (crash the craft). Re-arming is locked until RC link is available for at least 3 seconds and the arm switch (if used) is in the OFF position.
- `AUTO-LAND`: Enable an auto-level mode, center the flight sticks and set the throttle to a predefined value (`failsafe_throttle`) for a predefined time (`failsafe_off_delay`). This should allow the craft to come to a safer landing. Re-arming is locked until RC link is available for at least 30 seconds and the arm switch (if used) is in the OFF position.

`rx_min_usec`

The lowest channel value considered valid. e.g. PWM/PPM pulse length

`rx_max_usec`

The highest channel value considered valid. e.g. PWM/PPM pulse length

The `rx_min_usec` and `rx_max_usec` settings helps detect when your RX stops sending any data, enters failsafe mode or when the RX looses signal.

With a Graupner GR-24 configured for PWM output with failsafe on channels 1-4 set to OFF in the receiver settings then this setting, at its default value, will allow failsafe to be activated.

Testing

Bench test the failsafe system before flying - *remove props while doing so*.

1. Arm the craft.
2. Turn off transmitter or unplug RX.
3. Observe motors spin at configured throttle setting for configured duration.
4. Observe motors turn off after configured duration.
5. Ensure that when you turn on your TX again or reconnect the RX that you cannot re-arm once the motors have stopped.
6. Power cycle the FC.
7. Arm the craft.
8. Turn off transmitter or unplug RX.
9. Observe motors spin at configured throttle setting for configured duration.
10. Turn on TX or reconnect RX.
11. Ensure that your switch positions don't now cause the craft to disarm (otherwise it would fall out of the sky on regained signal).
12. Observe that normal flight behavior is resumed.
13. Disarm.

Field test the failsafe system.

1. Perform bench testing first!
2. On a calm day go to an unpopulated area away from buildings or test indoors in a safe controlled environment - e.g. inside a big net.
3. Arm the craft.
4. Hover over something soft (long grass, ferns, heather, foam, etc.).
5. Descend the craft and observe throttle position and record throttle value from your TX channel monitor. Ideally 1500 should be hover. So your value should be less than 1500.
6. Stop, disarm.

7. Set failsafe throttle to the recorded value.
8. Arm, hover over something soft again.
9. Turn off TX (!)
10. Observe craft descends and motors continue to spin for the configured duration.
11. Observe FC disarms after the configured duration.
12. Remove flight battery.

If craft descends too quickly then increase failsafe throttle setting.

Ensure that the duration is long enough for your craft to land at the altitudes you normally fly at.

Using a configured transmitter switch to activate failsafe mode, instead of switching off your TX, is good primary testing method in addition to the above procedure.

GPS

GPS features in Cleanflight are experimental. Please share your findings with the developers.

GPS works best if the GPS receiver is mounted above and away from other sources of interference.

The compass/mag sensor should be well away from sources of magnetic interference, e.g. keep it away from power wires, motors, ESCs.

Two GPS protocols are supported. NMEA text and UBLOX binary.

Configuration

Enable the GPS from the CLI as follows:

1. configure a serial port to use for GPS. (Serial.md)
2. set your GPS baud rate
3. enable the `feature GPS`
4. set the `gps_provider`
5. connect your GPS to the serial port configured for GPS.
6. save and reboot.

Note: GPS packet loss has been observed at 115200. Try using 57600 if you experience this.

For the connections step check the Board documentation for pins and port numbers.

GPS Provider

Set the `gps_provider` appropriately, for example `set gps_provider=UBLOX`

Value
NMEA
UBLOX

GPS Auto configuration

When using UBLOX it is a good idea to use GPS auto configuration so your FC gets the GPS messages it needs.

Enable GPS auto configuration as follows `set gps_auto_config=ON`.

If you are not using GPS auto configuration then ensure your GPS receiver sends out the correct messages at the right frequency. See below for manual UBlox settings.

SBAS

When using a UBLOX GPS the SBAS mode can be configured using `gps_sbas_mode`.

The default is AUTO.

Value	Region

AUTO	Global
EGNOS	Europe
WAAS	North America
MSAS	Asia
GAGAN	India

If you use a regional specific setting you may achieve a faster GPS lock than using AUTO.

This setting only works when `gps_auto_config=ON`

GPS Receiver Configuration

UBlox GPS units can either be configured using the FC or manually.

UBlox GPS manual configuration

Use UBox U-Center and connect your GPS to your computer. The CLI `gpspassthrough` command may be of use if you do not have a spare USART to USB adapter.

Note that many boards will not provide +5V from USB to the GPS module, such as the SPRacingF3; if you are using `gpspassthrough` you may need to connect a BEC to the controller if your board permits it, or use a standalone UART adapter. Check your board documentation to see if your GPS port is powered from USB.

Display the Packet Console (so you can see what messages your receiver is sending to your computer).

Display the Configation View.

Navigate to CFG (Configuration)

Select `Revert to default configuration`.
Click Send.

At this point you might need to disconnect and reconnect at the default baudrate - probably 9600 baud.

Navigate to PRT (Ports)

Set `Target` to `1 - Uart 1`
Set `Protocol In` to `0+1+2`
Set `Protocol Out` to `0+1`
Set `Buadrate` to `57600` `115200`
Press Send

This will immediatly "break" communication to the GPS. Since you haven't saved the new baudrate setting to the non-volatile memory you need to change the baudrate you communicate to the GPS without resetting the GPS. So `Disconnect`, Change baud rate to match, then `Connect`.

Click on PRT in the Configuration view again and inspect the packet console to make sure messages are being sent and acknowledged.

Next, to ensure the FC doesn't waste time processing unneeded messages, click on MSG and enable the following on UART1 alone with a rate of 1. When changing message target and rates remember to click Send after changing each message.:

```
NAV-POSLLH
NAV-DOP
NAV-SOL
NAV-VELNED
NAV-TIMEUTC
```

Enable the following on UART1 with a rate of 5, to reduce bandwidth and load on the FC.

```
NAV-SVINFO
```

All other message types should be disabled.

Next change the global update rate, click Rate (Rates) in the Configuration view.

Set Measurement period to 100 ms.
Set Navigation rate to 1.
Click Send.

This will cause the GPS receive to send the require messages out 10 times a second. If your GPS receiver cannot be set to use 100ms try 200ms (5hz) - this is less precise.

Next change the mode, click NAV5 (Navigation 5) in the Configuration View.

Set to Dynamic Model to Pedestrian and click Send.

Next change the SBAS settings. Click SBAS (SBAS Settings) in the Configuration View.

Set Subsystem to Enabled.
Set PRN Codes to Auto-Scan.
Click Send.

Finally, we need to save the configuration.

Click CFG (Configuration in the Configuration View.

Select Save current configuration and click Send.

UBlox Navigation model

Cleanflight will use Pedestrian when gps auto config is used.

From the UBlox documentation:

- Pedestrian - Applications with low acceleration and speed, e.g. how a pedestrian would move. Low acceleration assumed. MAX Altitude [m]: 9000, MAX Velocity [m/s]: 30, MAX Vertical, Velocity [m/s]: 20, Sanity check type: Altitude and Velocity, Max Position Deviation: Small.
- Portable - Applications with low acceleration, e.g. portable devices. Suitable for most situations. MAX Altitude [m]: 12000, MAX Velocity [m/s]: 310, MAX Vertical Velocity [m/s]: 50, Sanity check type: Altitude and Velocity, Max Position Deviation: Medium.
- Airborne < 1G - Used for applications with a higher dynamic range and vertical acceleration than a passenger car. No 2D position fixes supported. MAX Altitude [m]:

50000, MAX Velocity [m/s]: 100, MAX Vertical Velocity [m/s]: 100, Sanity check type: Altitude, Max Position Deviation: Large

Hardware

There are many GPS receivers available on the market.
Below are some examples of user-tested hardware.

Ublox

###U-Blox

NEO-M8

Module	Comments
U-blox Neo-M8N w/Compass	Pinout can be found in Pixfalcon manual. SDA and SCL can be attached to I2C bus for compass, TX and RX can be attached to UART for GPS. Power must be applied for either to function.
Reyax RY825AI	NEO-M8N, 18Hz UART USB interface GPS Glonass BeiDou QZSS antenna module flash. eBay (http://www.ebay.com/itm/RY825AI-18Hz-UART-USB-interface-GPS-Glonass-BeiDou-QZSS-antenna-module-flash/181566850426)

NEO-7

Module	Comments
U-blox Neo-7M w/Compass	HobbyKing (http://www.hobbyking.com/hobbyking/store/__55558__Ublox_Neo_7M_GPS_with_Co You have to set align_mag in the CLI to get the magnetometer working correctly: se forget to save.

NEO-6

Module	Comments
Ublox NEO-6M GPS with Compass	eBay (http://www.ebay.com/itm/111585855757)

Serial NMEA

MediaTek

Module	Comments
MTK 3329	Tested on hardware serial at 115200 baud (default) and on softserial at 19200 baud. The baudrate and refresh rate can be adjusted using the MiniGPS software (recommended if you lower the baudrate). The software will estimate the percentage of UART bandwidth used for your chosen baudrate and update rate.

G-Tune instructions.

The algorithm has been originally developed by Mohammad Hefny (mohammad.hefny@gmail.com):

> http://technicaladventure.blogspot.com/2014/06/zero-pids-tuner-for-multirotors.html
> http://diydrones.com/profiles/blogs/zero-pid-tunes-for-multirotors-part-2
> http://www.multiwii.com/forum/viewtopic.php?f=8&t=5190

The G-Tune functionality for Cleanflight is ported from the Harakiri firmware.

Safety preamble: *Use at your own risk*

The implementation you have here is quite different and just for adjusting the P values of ROLL/PITCH/YAW in Acro mode.
When flying in Acro mode (yaw tune in other modes possible as well - see below) you can activate G-Tune with an AUX box (switch) while the copter is armed.

It will start tuning the wanted / possible axis (see below) in a predefined range (see below).

After activation you will probably notice nothing! That means G-Tune will not start shaking your copter, you will have to do it (or simply fly and let it work).

The G-Tune is based on the gyro error so it is only active when you give no RC input (that would be an additional error). So if you just roll only pitch and yaw are tuned. If you stop rolling G-Tune will wait ca. 450ms to let the axis settle and then start tuning that axis again. All axis are treated independently.

The easiest way to tune all axis at once is to do some air-jumps with the copter in Acro (RC centered and G-Tune activated... of course..).

You can set a too high P for the axis as default in the GUI, when the copter starts shaking the wobbles will be detected and P tuned down (be careful with the strength setting though - see below).

Yaw tune is disabled in any copter with less than 4 motors (like tricopters).

G-Tune in Horizon or Level mode will just affect Yaw axis (if more than 3 motors...)

You will see the results in the GUI - the tuning results will only be saved if you enable G-Tune mode while the copter is disarmed and G-Tune was used before when armed. You also can save the configuration in an alternative way (like hitting save button in the GUI, casting an eepromwrite with trimming, acc calibration etc.)

TPA and G-Tune: It is not tested and will most likely not result into something good. However G-Tune might be able to replace TPA for you.

A typical use may go in this order:

1. Arm

2. Enable G-tune
3. Lift off slowly, avoid stick inputs (Roll, Pitch / Yaw).
4. Eventually the copter should fly well. Perhaps do a few throttle punch outs and fly around a bit. Take note if each punch out seems to become smoother with less oscillation and the overall flight performance.
5. Disable G-tune
6. Land
7. Disarm, but don’t power off.
8. If these are desired results then either a) Connect cleanflight configurator review and save the configuration. or b) Enable G-Tune again to save settings.
9. Power off.

If the results are not desired look into changing the parameters as shown below and try again.

Some other notes and instructions can be found here:

> http://www.rcgroups.com/forums/showpost.php?p=31321635&postcount=6160
> http://www.rcgroups.com/forums/showpost.php?p=31525114&postcount=7150

Parameters and their function:

```
gtune_loP_rll        = 10  [0..200] Lower limit of ROLL P during G-Tune.  Note
"10" means "1.0" in the GUI.
gtune_loP_ptch       = 10  [0..200] Lower limit of PITCH P during G-Tune. Note
"10" means "1.0" in the GUI.
gtune_loP_yw         = 10  [0..200] Lower limit of YAW P during G-Tune.   Note
"10" means "1.0" in the GUI.
gtune_hiP_rll        = 100 [0..200] Higher limit of ROLL P during G-Tune. 0
Disables tuning for that axis.  Note "100" means "10.0" in the GUI.
gtune_hiP_ptch       = 100 [0..200] Higher limit of PITCH P during G-Tune. 0
Disables tuning for that axis. Note "100" means "10.0" in the GUI.
gtune_hiP_yw         = 100 [0..200] Higher limit of YAW P during G-Tune. 0
Disables tuning for that axis.   Note "100" means "10.0" in the GUI.
gtune_pwr            = 0   [0..10] Strength of adjustment
gtune_settle_time    = 450 [200..1000] Settle time in ms
gtune_average_cycles = 16  [8..128] Number of looptime cycles used for gyro
average calcullation
```

So you have lower and higher limits for each P for every axis. The preset range (GUI: 1.0 - 10.0) is quiet broad to represent most setups.

If you want tighter or more loose ranges change them here. gtune_loP_XXX can be configured lower than "10" that means a P of "1.0" in the GUI. So you can have "Zero P" but you may get sluggish initial control.

If you want to exclude one axis from the tuning you must set gtune_hiP_XXX to zero. Let's say you want to disable yaw tuning write in CLI `set gtune_hiP_yw = 0`. Note: The MultiWii Wiki advises you to trim the yaw axis on your transmitter. If you have done so (yaw not neutral on your RC) yaw tuning will be disabled.

You can adjust the strength of tuning by using `set gtune_pwr = N`. My small copter works fine with 0 and doesn't like a value of "3". My big copter likes "gtune_pwr = 5". It shifts the tuning to higher values and if too high can diminish the wobble blocking! So start with 0 (default). If you feel your resulting P is always too low for you, increase gtune_pwr. You will see it getting a little shaky if value is too high.

Installation

Using the configurator

This is a generic procedure to flash a board using the configurator. The configurator does not yet support all boards, so please check the documentation corresponding to your board before proceeding.

Make sure you have the Cleanflight Configurator (https://github.com/cleanflight/cleanflight-configurator) installed, then:

- Connect the flight controller to the PC.
- Start the Cleanflight Configurator.
- Click on "Disconnect" if the configurator connected to the board automatically.
- Click on the "Firmware Flasher" tab.
- Make sure you have internet connectivity and click on the "Load Firmware [Online]" button.
- Click on the "Choose a Firmware / Board" dropdown menu, and select the latest stable version for your flight controller.
- IMPORTANT: Read and understand the release notes that are displayed. When upgrading review all release notes since your current firmware.
- If this is the first time Cleanflight is flashed to the board, tick the "Full Chip Erase" checkbox.
- Connect the flight controller board to the PC. Ensure the correct serial port is selected.
- Click on the "Flash Firmware" button and hold still (do not breathe, too).
- When the progress bar becomes green and reads "Programming: SUCCESSFUL" you are done!

Manually

See the board specific flashing instructions.

Upgrading

When upgrading be sure to backup / dump your existing settings. Some firmware releases are not backwards compatible and default settings are restored when the FC detects an out of date configuration.

Backup/Restore process

See the CLI section of the docs for details on how to backup and restore your configuration via the CLI.

Cleanflight

Welcome to CleanFlight!

Cleanflight is an community project which attempts to deliver flight controller firmware and related tools.

Primary Goals

- Community driven.
- Friendly project atmosphere.
- Focus on the needs of users.
- Great flight performance.
- Understandable and maintainable code.

Hardware

See the flight controller hardware (Boards.md) chapter for details.

Software

There are two primary components, the firmware and the configuration tool. The firmware is the code that runs on the flight controller board. The GUI configuration tool (configurator) is used to configure the flight controller, it runs on Windows, OSX and Linux.

Feedback & Contributing

We welcome all feedback. If you love it we want to hear from you, if you have problems please tell us how we could improve things so we can make it better for everyone.

If you want to contribute please see the notes here:

https://github.com/cleanflight/cleanflight#contributing

Developers should read this:

https://github.com/cleanflight/cleanflight/blob/master/CONTRIBUTING.md

LED Strip

Cleanflight supports the use of addressable LED strips. Addressable LED strips allow each LED in the strip to
be programmed with a unique and independant color. This is far more advanced than the normal RGB strips which
require that all the LEDs in the strip show the same color.

Addressable LED strips can be used to show information from the flight controller system, the current implementation
supports the following:

- Up to 32 LEDs.
- Indicators showing pitch/roll stick positions.
- Heading/Orientation lights.
- Flight mode specific color schemes.
- Low battery warning.
- AUX operated on/off switch.
- GPS state.
- RSSI level.
- Battery level.

Support for more than 32 LEDs is possible, it just requires additional development.

Supported hardware

Only strips of 32 WS2811/WS2812 LEDs are supported currently. If the strip is longer than 32 LEDs it does not matter,
but only the first 32 are used.

WS2812 LEDs require an 800khz signal and precise timings and thus requires the use of a dedicated hardware timer.

Note: Not all WS2812 ICs use the same timings, some batches use different timings.

It could be possible to be able to specify the timings required via CLI if users request it.

Tested Hardware

- Adafruit NeoPixel Jewel 7 (https://www.adafruit.com/products/2226) (preliminary testing)
    - Measured current consumption in all white mode ~ 350 mA.
    - Fits well under motors on mini 250 quads.
- Adafruit NeoPixel Stick (https://www.adafruit.com/products/1426) (works well)
    - Measured current consumption in all white mode ~ 350 mA.

WS2811 vs WS2812

The WS2811 (https://cdn-shop.adafruit.com/datasheets/WS2811.pdf) is a LED driver IC which is connected to an RGB LED. It accepts data in the form of 8 bits each of Red-Green-Blue.

The WS2812 (https://cdn-shop.adafruit.com/datasheets/WS2812.pdf) is integrated into the package of a 50:50 LED rather than as a separate device. It accepts data in the form of 8 bits each of Green-Red-Blue.

It is thus possible, depending on the LED board/strip being used that either Red-Green-Blue or Green-Red-Blue encoding may be required. This may be controlled by setting the following.

```
set ledstrip_grb_rgb = RGB
```

or

```
set ledstrip_grb_rgb = GRB
```

Then confirm the required setting by simply setting an LED to be green. If it lights up red, you have the wrong setting.

Connections

WS2812 LED strips generally require a single data line, 5V and GND.

WS2812 LEDs on full brightness can consume quite a bit of current. It is recommended to verify the current draw and ensure your
supply can cope with the load. On a multirotor that uses multiple BEC ESC's you can try use a different BEC to the one the FC
uses. e.g. ESC1/BEC1 -> FC, ESC2/BEC2 -> LED strip. It's also possible to power one half of the strip from one BEC and the other half
from another BEC. Just ensure that the GROUND is the same for all BEC outputs and LEDs.

Target	Pin	LED Strip Signal
Naze	RC5	Data In PA6
CC3D	RCO5	Data In PB4
ChebuzzF3/F3Discovery	PB8	Data In PB8
Sparky	PWM5	Data In PA6

Since RC5 is also used for SoftSerial on the Naze it means that you cannot use SoftSerial and led strips at the same time.
Additionally, since RC5 is also used for Parallel PWM RC input on both the Naze, Chebuzz and STM32F3Discovery targets, led strips
can not be used at the same time at Parallel PWM.

If you have LEDs that are intermittent, flicker or show the wrong colors then drop the VIN to less than 4.7v, e.g. by using an inline
diode on the VIN to the LED strip. The problem occurs because of the difference in voltage between the data signal and the power
signal. The WS2811 LED's require the data signal (Din) to be between 0.3 * Vin (Max) and 0.7 * VIN (Min) to register valid logic
low/high signals. The LED pin on the CPU will always be between 0v to ~3.3v, so the Vin should be 4.7v (3.3v / 0.7 = 4.71v).
Some LEDs are more tolerant of this than others.

The datasheet can be found here: http://www.adafruit.com/datasheets/WS2812.pdf

Configuration

The led strip feature can be configured via the GUI.

GUI:
Enable the Led Strip feature via the GUI under setup.

Configure the leds from the Led Strip tab in the cleanflight GUI.
First setup how the led's are laid out so that you can visualize it later as you configure and so the flight controller knows how many led's there are available.

There is a step by step guide on how to use the GUI to configure the Led Strip feature using the GUI http://blog.oscarliang.net/setup-rgb-led-cleanflight/ which was published early 2015 by Oscar Liang which may or may not be up-to-date by the time you read this.

CLI:
Enable the `LED_STRIP` feature via the cli:

```
feature LED_STRIP
```

If you enable LED_STRIP feature and the feature is turned off again after a reboot then check your config does not conflict with other features, as above.

Configure the LEDs using the `led` command.

The `led` command takes either zero or two arguments - an zero-based led number and a sequence which indicates pair of coordinates, direction flags and mode flags and a color.

If used with zero arguments it prints out the led configuration which can be copied for future reference.

Each led is configured using the following template: `x,y:ddd:mmm:cc`

`x` and `y` are grid coordinates of a 0 based 16x16 grid, north west is 0,0, south east is 15,15
`ddd` specifies the directions, since an led can face in any direction it can have multiple directions. Directions are:

`N` - North
`E` - East
`S` - South
`W` - West
`U` - Up
`D` - Down

For instance, an LED that faces South-east at a 45 degree downwards angle could be configured as `SED`.

Note: It is perfectly possible to configure an LED to have all directions `NESWUD` but probably doesn't make sense.

`mmm` specifies the modes that should be applied an LED.

Each LED has one base function:

- C - Color.
- F - Flight mode & Orientation
- A - Armed state.
- R - Ring thrust state.
- G - GPS state.
- S - RSSSI level.
- L - Battery Level.

And each LED has overlays:

- W - Wwarnings.
- I - Indicator.
- T - Thrust state.
- B - Blink (flash twice) mode.
- O - Lars0n Scanner (Cylon Effect).
- N - Blink on laNding (throttle < 50%).

cc specifies the color number (0 based index).

Example:

```
led 0 0,15:SD:AWI:0
led 1 15,0:ND:AWI:0
led 2 0,0:ND:AWI:0
led 3 0,15:SD:AWI:0
led 4 7,7::C:1
led 5 8,8::C:2
led 6 8,9::B:1
```

To erase an led, and to mark the end of the chain, use `0,0::` as the second argument, like this:

```
led 4 0,0:::
```

It is best to erase all LEDs that you do not have connected.

Modes

Warning

This mode simply uses the LEDs to flash when warnings occur.

Warning	LED Pattern	Notes
Arm-lock enabled	flash between green and off	occurs calibration or when unarmed and the aircraft is tilted too much
Low Battery	flash red and off	battery monitoring must be enabled. May trigger temporarily under high-throttle due to voltage drop
Failsafe	flash between light blue and yellow	Failsafe must be enabled

Flash patterns appear in order, so that it's clear which warnings are enabled.

GPS state

This mode shows the GPS state and satellite count.

No fix = red LED
3D fix = green LED

The LEDs will blink as many times as the satellite count, then pause and start again.

RSSI level

This mode binds the LED color to RSSI level.

Color	RSSI
Green	100%
Lime green	80%
Yellow	60%
Orange	40%
Red	20%
Deep pink	0%

When RSSI is below 50% is reached, LEDs will blink slowly, and they will blink fast when under 20%.

Battery level

This mode binds the LED color to remaining battery capacity.

Color	Capacity
Green	100%
Lime green	80%
Yellow	60%
Orange	40%
Red	20%
Deep pink	0%

When Warning or Critial voltage is reached, LEDs will blink slowly or fast.
Note: this mode requires a current sensor. If you don't have the actual device you can set up a virtual current sensor (see Battery (Battery.md)).

Blink

This mode blinks the current LED, alternatively from black to the current active color.

Blink on landing

This mode blinks the current LED, alternatively from black to the current active color, when throttle is below 50% and the craft is armed.

Larson Scanner (Cylon Effect)

The Larson Scanner replicates the scanning "eye" effect seen on the mechanical Cylons and on Kitt from Knight Rider.

This overlay merely varies the brightness of each LED's current color.

Flight Mode & Orientation

This mode shows the flight mode and orientation.

When flight modes are active then the LEDs are updated to show different colors depending on the mode, placement on the grid and direction.

LEDs are set in a specific order:

- LEDs that marked as facing up or down.
- LEDs that marked as facing west or east AND are on the west or east side of the grid.
- LEDs that marked as facing north or south AND are on the north or south side of the grid.

That is, south facing LEDs have priority.

The mapping between modes led placement and colors is currently fixed and cannot be changed.

Indicator

This mode flashes LEDs that correspond to roll and pitch stick positions. i.e. they indicate the direction the craft is going to turn.

Mode	Direction	LED Color
Orientation	North	WHITE
Orientation	East	DARK VIOLET
Orientation	South	RED
Orientation	West	DEEP PINK
Orientation	Up	BLUE
Orientation	Down	ORANGE
Head Free	North	LIME GREEN
Head Free	East	DARK VIOLET
Head Free	South	ORANGE
Head Free	West	DEEP PINK
Head Free	Up	BLUE
Head Free	Down	ORANGE
Horizon	North	BLUE
Horizon	East	DARK VIOLET
Horizon	South	YELLOW
Horizon	West	DEEP PINK
Horizon	Up	BLUE
Horizon	Down	ORANGE
Angle	North	CYAN
Angle	East	DARK VIOLET
Angle	South	YELLOW
Angle	West	DEEP PINK
Angle	Up	BLUE
Angle	Down	ORANGE

Mag	North	MINT GREEN
Mag	East	DARK VIOLET
Mag	South	ORANGE
Mag	West	DEEP PINK
Mag	Up	BLUE
Mag	Down	ORANGE
Baro	North	LIGHT BLUE
Baro	East	DARK VIOLET
Baro	South	RED
Baro	West	DEEP PINK
Baro	Up	BLUE
Baro	Down	ORANGE

Armed state

This mode toggles LEDs between green and blue when disarmed and armed, respectively.

Note: Armed State cannot be used with Flight Mode.

Thrust state

This mode fades the LED current LED color to the previous/next color in the HSB color space depending on throttle stick position. When the
throttle is in the middle position the color is unaffected, thus it can be mixed with orientation colors to indicate orientation and throttle at
the same time. Thrust should normally be combined with Color or Mode/Orientation.

Thrust ring state

This mode is allows you to use one or multiple led rings (e.g. NeoPixel ring) for an afterburner effect. The light pattern rotates clockwise as throttle increases.

A better effect is acheived when LEDs configured for thrust ring have no other functions.

LED direction and X/Y positions are irrelevant for thrust ring LED state. The order of the LEDs that have the state determines how the LED behaves.

Each LED of the ring can be a different color. The color can be selected between the 16 colors availables.

For example, led 0 is set as a Ring thrust state led in color 13 as follow.

```
led 0 2,2::R:13
```

LED strips and rings can be combined.

Solid Color

The mode allows you to set an LED to be permanently on and set to a specific color.

x,y position and directions are ignored when using this mode.

Other modes will override or combine with the color mode.

For example, to set led 0 to always use color 10 you would issue this command.

```
led 0 0,0::C:10
```

Colors

Colors can be configured using the cli `color` command.

The `color` command takes either zero or two arguments - an zero-based color number and a sequence which indicates pair of hue, saturation and value (HSV).

See http://en.wikipedia.org/wiki/HSL_and_HSV

If used with zero arguments it prints out the color configuration which can be copied for future reference.

The default color configuration is as follows:

Index	Color
0	black
1	white
2	red
3	orange
4	yellow
5	lime green
6	green
7	mint green
8	cyan
9	light blue
10	blue
11	dark violet
12	magenta
13	deep pink
14	black
15	black

```
color 0 0,0,0
color 1 0,255,255
color 2 0,0,255
color 3 30,0,255
color 4 60,0,255
color 5 90,0,255
color 6 120,0,255
color 7 150,0,255
color 8 180,0,255
color 9 210,0,255
color 10 240,0,255
color 11 270,0,255
color 12 300,0,255
color 13 330,0,255
color 14 0,0,0
color 15 0,0,0
```

Mode Colors Assignement

Mode Colors can be configured using the cli `mode_color` command.

- No arguments: lists all mode colors
- arguments: mode, function, color

First 6 groups of ModeIndexes are :

mode	name
0	orientation
1	headfree
2	horizon
3	angle
4	mag
5	baro
6	special

Modes 0 to 5 functions:

function	name
0	north
1	east
2	south
3	west
4	up
5	down

Mode 6 use these functions:

function	name
0	disarmed
1	armed
2	animation

3	background
4	blink background
5	gps: no satellites
6	gps: no fix
7	gps: 3D fix

The ColorIndex is picked from the colors array ("palette").

Examples (using the default colors):

- set armed color to red: `mode_color 6 1 2`
- set disarmed color to yellow: `mode_color 6 0 4`
- set Headfree mode 'south' to Cyan: `mode_color 1 2 8`

Positioning

Cut the strip into sections as per diagrams below. When the strips are cut ensure you reconnect each output to each input with cable where the break is made.
e.g. connect 5V out to 5V in, GND to GND and Data Out to Data In.

Orientation is when viewed with the front of the aircraft facing away from you and viewed from above.

Example 12 LED config

The default configuration is as follows

```
led 0 15,15:ES:IA:0
led 1 15,8:E:WF:0
led 2 15,7:E:WF:0
led 3 15,0:NE:IA:0
led 4 8,0:N:F:0
led 5 7,0:N:F:0
led 6 0,0:NW:IA:0
led 7 0,7:W:WF:0
led 8 0,8:W:WF:0
led 9 0,15:SW:IA:0
led 10 7,15:S:WF:0
led 11 8,15:S:WF:0
led 12 7,7:U:WF:0
led 13 8,7:U:WF:0
led 14 7,8:D:WF:0
led 15 8,8:D:WF:0
led 16 8,9::R:3
led 17 9,10::R:3
led 18 10,11::R:3
led 19 10,12::R:3
led 20 9,13::R:3
led 21 8,14::R:3
led 22 7,14::R:3
led 23 6,13::R:3
led 24 5,12::R:3
led 25 5,11::R:3
led 26 6,10::R:3
led 27 7,9::R:3
led 28 0,0:::0
led 29 0,0:::0
led 30 0,0:::0
led 31 0,0:::0
```

Which translates into the following positions:

```
    6             3
     \           /
      \   5-4   /
       \ FRONT /
   7,8 | 12-15 | 1,2
       /  BACK \
      /  10,11  \
     /           \
    9             0
      RING 16-27
```

LEDs 0,3,6 and 9 should be placed underneath the quad, facing downwards.
LEDs 1-2, 4-5, 7-8 and 10-11 should be positioned so the face east/north/west/south, respectively.
LEDs 12-13 should be placed facing down, in the middle
LEDs 14-15 should be placed facing up, in the middle
LEDs 16-17 should be placed in a ring and positioned at the rear facing south.

This is the default so that if you don't want to place LEDs top and bottom in the middle just connect the first 12 LEDs.

Example 16 LED config

```
led 0 15,15:SD:IA:0
led 1 8,8:E:FW:0
led 2 8,7:E:FW:0
led 3 15,0:ND:IA:0
led 4 7,7:N:FW:0
led 5 8,7:N:FW:0
led 6 0,0:ND:IA:0
led 7 7,7:W:FW:0
led 8 7,8:W:FW:0
led 9 0,15:SD:IA:0
led 10 7,8:S:FW:0
led 11 8,8:S:FW:0
led 12 7,7:D:FW:0
led 13 8,7:D:FW:0
led 14 7,7:U:FW:0
led 15 8,7:U:FW:0
```

Which translates into the following positions:

```
    6             3
     \           /
      \   5-4   /
     7 \ FRONT / 2
       | 12-15 |
     8 /  BACK \ 1
      /  10-11  \
     /           \
    9             0
```

LEDs 0,3,6 and 9 should be placed underneath the quad, facing downwards.
LEDs 1-2, 4-5, 7-8 and 10-11 should be positioned so the face east/north/west/south, respectively.
LEDs 12-13 should be placed facing down, in the middle
LEDs 14-15 should be placed facing up, in the middle

Exmple 28 LED config

```
#right rear cluster
led 0 9,9:S:FWT:0
led 1 10,10:S:FWT:0
led 2 11,11:S:IA:0
led 3 11,11:E:IA:0
led 4 10,10:E:AT:0
led 5 9,9:E:AT:0
# right front cluster
led 6 10,5:S:F:0
led 7 11,4:S:F:0
led 8 12,3:S:IA:0
led 9 12,2:N:IA:0
led 10 11,1:N:F:0
led 11 10,0:N:F:0
# center front cluster
led 12 7,0:N:FW:0
led 13 6,0:N:FW:0
led 14 5,0:N:FW:0
led 15 4,0:N:FW:0
# left front cluster
led 16 2,0:N:F:0
led 17 1,1:N:F:0
led 18 0,2:N:IA:0
led 19 0,3:W:IA:0
led 20 1,4:S:F:0
led 21 2,5:S:F:0
# left rear cluster
led 22 2,9:W:AT:0
led 23 1,10:W:AT:0
led 24 0,11:W:IA:0
led 25 0,11:S:IA:0
led 26 1,10:S:FWT:0
led 27 2,9:S:FWT:0
```

```
       16-18  9-11
19-21 \           / 6-8
       \  12-15  /
        \ FRONT /
        /  BACK \
       /         \
22-24 /           \ 3-5
       25-27   0-2
```

All LEDs should face outwards from the chassis in this configuration.

Note:
This configuration is specifically designed for the [Alien Spider AQ50D PRO 250mm frame (http://www.goodluckbuy.com/alien-spider-aq50d-pro-250mm-mini-quadcopter-carbon-fiber-micro-multicopter-frame.html)](http://www.goodluckbuy.com/alien-spider-aq50d-pro-250mm-mini-quadcopter-carbon-fiber-micro-multicopter-frame.html).

Troubleshooting

On initial power up the LEDs on the strip will be set to WHITE. This means you can attach a current meter to verify
the current draw if your measurement equipment is fast enough. Most 5050 LEDs will draw 0.3 Watts a piece.
This also means that you can make sure that each R,G and B LED in each LED module on the strip is also functioning.

After a short delay the LEDs will show the unarmed color sequence and or low-battery warning sequence.

Also check that the feature `LED_STRIP` was correctly enabled and that it does not conflict with other features, as above.

Mixer

Cleanflight supports a number of mixing configurations as well as custom mixing. Mixer configurations determine how the servos and motors work together to control the aircraft.

Configuration

To use a built-in mixing configuration, you can use the Chrome configuration GUI. It includes images of the various mixer types to assist in making the proper connections. See the Configuration section of the documentation for more information on the GUI.

You can also use the Command Line Interface (CLI) to set the mixer type:

1. Use `mixer list` to see a list of supported mixes
2. Select a mixer. For example, to select TRI, use `mixer TRI`
3. You must use save to preserve your changes

Supported Mixer Types

Name	Description	Motors	Servos
TRI	Tricopter	M1-M3	S1
QUADP	Quadcopter-Plus	M1-M4	None
QUADX	Quadcopter-X	M1-M4	None
BI	Bicopter (left/right)	M1-M2	S1, S2
GIMBAL	Gimbal control	N/A	S1, S2
Y6	Y6-copter	M1-M6	None
HEX6	Hexacopter-Plus	M1-M6	None
FLYING_WING	Fixed wing; elevons	M1	S1, S2
Y4	Y4-copter	M1-M4	None
HEX6X	Hexacopter-X	M1-M6	None
OCTOX8	Octocopter-X (over/under)	M1-M8	None
OCTOFLATP	Octocopter-FlatPlus	M1-M8	None
OCTOFLATX	Octocopter-FlatX	M1-M8	None
AIRPLANE	Fixed wing; Ax2, R, E	M1	S1, S2, S3, S4
HELI_120_CCPM			
HELI_90_DEG			
VTAIL4	Quadcopter with V-Tail	M1-M4	N/A
HEX6H	Hexacopter-H	M1-M6	None
PPM_TO_SERVO			
DUALCOPTER	Dualcopter	M1-M2	S1, S2
SINGLECOPTER	Conventional helicopter	M1	S1
ATAIL4	Quadcopter with A-Tail	M1-M4	N/A
CUSTOM	User-defined		
CUSTOM AIRPLANE	User-defined airplane	M1-M2	S1-S8
CUSTOM TRICOPTER	User-defined tricopter		

Servo configuration

The cli `servo` command defines the settings for the servo outputs.
The cli mixer `smix` command controls how the mixer maps internal FC data (RC input, PID stabilization output, channel forwarding, etc) to servo outputs.

Channel Forwarding

Channel Forwarding allows you to forward your AUX channels directly to servos over PWM pins 5-8. You can enable it under features in the GUI or using the cli with `feature CHANNEL_FORWARDING`. This requires you to run PPM or another serial RC protocol, and is currently supported on NAZE and SPRACINGF3 targets.
Note that if you have the led feature enabled on the NAZE target, AUX1-2 is mapped to PWM13-14 instead. So for instance if you enable this feature on a Naze AUX1 from your receiver will automatically be forwarded to PWM5 as a servo signal.

cli servo

`servo <min> <max> <middle> <angleMin> <angleMax> <rate> <forwardFromChannel>`

- `<min>`, `<max>` - limit servo travel, in uS
- `<middle>` - mid value when not forwarding, value from servo mixer is added to this.
- `<angleMin>`, `<angleMax>` - unused
- `<rate>` - scale for value from servo mixer or gimbal input, -100% .. 100%
- `<forwardFromChannel>` - use RC channel value as reference instead of `<middle>`. Servo will follow given RC channel, with possible correction from servo mixer. `<min>`, `<max>` are still honored.

Servo filtering

A low-pass filter can be enabled for the servos. It may be useful for avoiding structural modes in the airframe, for example.

Configuration

Currently it can only be configured via the CLI:

1. Use `set servo_lowpass_freq = nnn` to select the cutoff frequency. Valid values range from 10Hz to 400Hz, second order filter is used.
2. Use `set servo_lowpass_enable = ON` to enable filtering.

Tuning

One method for tuning the filter cutoff is as follows:

1. Ensure your vehicle can move at least somewhat freely in the troublesome axis. For example, if you are having yaw oscillations on a tricopter, ensure that the copter is supported in a way that allows it to rotate left and right to at least some degree.

Suspension near the CG is ideal. Alternatively, you can just fly the vehicle and trigger the problematic condition you are trying to eliminate, although tuning will be more tedious.

2. Tap the vehicle at its end in the axis under evaluation. Directly commanding the servo in question to move may also be used. In the tricopter example, tap the end of the tail boom from the side, or command a yaw using your transmitter.
3. If your vehicle oscillates for several seconds or even continues oscillating indefinitely, then the filter cutoff frequency should be reduced. Reduce the value of `servo_lowpass_freq` by half its current value and repeat the previous step.
4. If the oscillations are dampened within roughly a second or are no longer present, then you are done. Be sure to run `save`.

Custom Motor Mixing

Custom motor mixing allows for completely customized motor configurations. Each motor must be defined with a custom mixing table for that motor. The mix must reflect how close each motor is with reference to the CG (Center of Gravity) of the flight controller. A motor closer to the CG of the flight controller will need to travel less distance than a motor further away.

Steps to configure custom mixer in the CLI:

1. Use `mixer custom` to enable the custom mixing.
2. Use `mmix reset` to erase the any existing custom mixing.
3. Optionally use `mmix load` <name> to start with one of available mixers.
4. Issue a `mmix` statement for each motor.

The `mmix` statement has the following syntax: `mmix n THROTTLE ROLL PITCH YAW`

Mixing table parameter	Definition
n	Motor ordering number
THROTTLE	Indicates how much throttle is mixed for this motor. All values used in current configurations are set to 1.0 (full throttle mixing), but other non-zero values may be used. Unused set to 0.0.
ROLL	Indicates how much roll authority this motor imparts to the roll of the flight controller. Accepts values nominally from -1.0 to 1.0.
PITCH	Indicates the pitch authority this motor has over the flight controller. Also accepts values nominally from -1.0 to 1.0.
YAW	Indicates the direction of the motor rotation in relationship with the flight controller. 1.0 = CCW -1.0 = CW.

Note: the `mmix` command may show a motor mix that is not active, custom motor mixes are only active for models that use custom mixers.

Note: You have to configure every motor number starting at 0. Your command will be ignored if there was no `mmix` command for the previous motor number (mixer stops on first THROTTLE value that is zero). See example 5.

Custom Servo Mixing

Custom servo mixing rules can be applied to each servo. Rules are applied in the order they are defined.

`smix`

Prints current servo mixer

Note: the `smix` command may show a servo mix that is not active, custom servo mixes are only active for models that use custom mixers.

`smix reset`

Erase custom mixer. Servo reversal in current profile ONLY is erased too.

`smix load <name>`

Load servo part of given configuration (<name> is from `mixer list`)

`smix <rule> <servo> <source> <rate> <speed> <min> <max> <box>`

- <rule> is index of rule, used mainly for bookkeeping. Rules are applied in this order, but ordering has no influence on result in current code.
- <servo>

id	Servo slot
0	GIMBAL PITCH
1	GIMBAL ROLL
2	ELEVATOR / SINGLECOPTER_4
3	FLAPPERON 1 (LEFT) / SINGLECOPTER_1
4	FLAPPERON 2 (RIGHT) / BICOPTER_LEFT / DUALCOPTER_LEFT / SINGLECOPTER_2
5	RUDDER / BICOPTER_RIGHT / DUALCOPTER_RIGHT / SINGLECOPTER_3
6	THROTTLE (Based ONLY on the first motor output)
7	FLAPS

Only some <servo> channels are connected to output, based on mode. For custom modes:

- RUDDER for CUSTOM_TRI
- ELEVATOR ... FLAPS for CUSTOM_AIRPLANE
- no servos for CUSTOM

GIMBAL handling is hard-coded, mmix rule is ignored.

- <source>

id	Input sources
0	Stabilized ROLL
1	Stabilized PITCH
2	Stabilized YAW
3	Stabilized THROTTLE (ONLY the first motor output)
4	RC ROLL
5	RC PITCH

6 RC YAW
7 RC THROTTLE
8 RC AUX 1
9 RC AUX 2
10 RC AUX 3
11 RC AUX 4
12 GIMBAL PITCH
13 GIMBAL ROLL

Stabilized ROLL/PITCH/YAW is taken directly from RC command when in PASSTHRU mode.

- <rate> is used to scale <source>, -100% - 100% is allowed. Note that servo reversal may be applied, see below. Zero <rate> will terminate smix table.
- <speed> will limit <source> speed when non-zero. This speed is taken per-rule, so you may limit only some sources. Value is maximal change of value per loop (1ms with default configuration)
- <min> <max> - Value in percentage of full servo range. For symmetrical servo limits (equal distance between mid and min/max), 0% is servo min, 50% is servo center, 100% is max servo position. When mid position is asymmetrical, 0% and 100% limits will be shifted.
- <box> rule will be applied only when <box> is zero or corresponding SERVOx mode is enabled.

smix reverse

Print current servo reversal configuration

smix reverse <servo> <source> r|n

Each <source> may be reversed or normal for given <servo>. It is almost equivalent to using negative <rate> in given rule, but <min>,<max>limits are applied to value before reversing.smix reverse` works for non-custom mixers too.

e.g. when using the TRI mixer to reverse the tail servo on a tricopter use this:

smix reverse 5 2 r

i.e. when mixing rudder servo slot (5) using Stabilized YAW input source (2) reverse the direction (r)

smix reverse is a per-profile setting. So ensure you configure it for your profiles as required.

Example 1: A KK2.0 wired motor setup

Here's an example of a X configuration quad, but the motors are still wired using the KK board motor numbering scheme.

```
KK2.0 Motor Layout

  1CW      2CCW
     \    /
       KK
     /    \
  4CCW     3CW
```

1. Use `mixer custom`
2. Use `mmix reset`
3. Use `mmix 0 1.0, 1.0, -1.0, -1.0` for the Front Left motor. It tells the flight controller the #1 motor is used, provides positive roll, provides negative pitch and is turning CW.
4. Use `mmix 1 1.0, -1.0, -1.0, 1.0` for the Front Right motor. It still provides a negative pitch authority, but unlike the front left, it provides negative roll authority and turns CCW.
5. Use `mmix 2 1.0, -1.0, 1.0, -1.0` for the Rear Right motor. It has negative roll, provides positive pitch when the speed is increased and turns CW.
6. Use `mmix 3 1.0, 1.0, 1.0, 1.0` for the Rear Left motor. Increasing motor speed imparts positive roll, positive pitch and turns CCW.

Example 2: A HEX-U Copter

Here is an example of a U-shaped hex; probably good for herding giraffes in the Sahara. Because the 1 and 6 motors are closer to the roll axis, they impart much less force than the motors mounted twice as far from the FC CG. The effect they have on pitch is the same as the forward motors because they are the same distance from the FC CG. The 2 and 5 motors do not contribute anything to pitch because speeding them up and slowing them down has no effect on the forward/back pitch of the FC.

```
HEX6-U

.4........3.
............
.5...FC...2.
............
...6....1...
```

Command	Roll	Pitch	Yaw
Use `mmix 0 1.0, -0.5, 1.0, -1.0`	half negative	full positive	CW
Use `mmix 1 1.0, -1.0, 0.0, 1.0`	full negative	none	CCW
Use `mmix 2 1.0, -1.0, -1.0, -1.0`	full negative	full negative	CW
Use `mmix 3 1.0, 1.0, -1.0, 1.0`	full positive	full negative	CCW
Use `mmix 4 1.0, 1.0, 0.0, -1.0`	full positive	none	CW
Use `mmix 5 1.0, 0.5, 1.0, 1.0`	half positive	full positive	CCW

Example 3: Custom tricopter

```
mixer CUSTOMTRI
mmix reset
mmix 0 1.000 0.000 1.333 0.000
mmix 1 1.000 -1.000 -0.667 0.000
mmix 2 1.000 1.000 -0.667 0.000
smix reset
smix 0 5 2 100 0 0 100 0
profile 0
smix reverse 5 2 r
profile 1
smix reverse 5 2 r
profile 2
smix reverse 5 2 r
```

Example 4: Custom Airplane with Differential Thrust

Here is an example of a custom twin engine plane with Differential Thrust (http://rcvehicles.about.com/od/rcairplanes/ss/RCAirplaneBasic.htm#step8)
Motors take the first 2 pins, the servos take pins as indicated in the [Servo slot] chart above. Settings bellow have motor yaw influence at "0.3", you can change this number to have more or less differential thrust over the two motors.
Note: You can look at the Motors tab in Cleanflight Cofigurator (https://chrome.google.com/webstore/detail/cleanflight-configurator/enacoimjcgeinfnnnpajinjgmkahmfgb?hl=en) to see motor and servo outputs.

Pins	Outputs
1	Left Engine
2	Right Engine
3	Pitch / Elevator
4	Roll / Aileron
5	Roll / Aileron
6	Yaw / Rudder
7	[EMPTY]
8	[EMPTY]

```
mixer CUSTOMAIRPLANE
mmix reset
mmix 0 1.0 0.0 0.0 0.3   # Left Engine
mmix 1 1.0 0.0 0.0 -0.3  # Right Engine

smix reset
# Rule  Servo   Source  Rate    Speed   Min Max Box
smix 0 3 0 100 0 0 100 0  # Roll / Aileron
smix 1 4 0 100 0 0 100 0  # Roll / Aileron
smix 2 5 2 100 0 0 100 0  # Yaw / Rudder
smix 3 2 1 100 0 0 100 0  # Pitch / Elevator
```

Example 5: Use motor output 0,1,2,4 because your output 3 is broken

For this to work you have to make a dummy mmix for motor 3. We do this by just saying it has 0 impact on yaw, roll and pitch.

```
mixer custom
mmix reset
mmix 0 1.0, -1.0, 1.0, -1.0
mmix 1 1.0, -1.0, -1.0,  1.0
mmix 2 1.0, 1.0, 1.0, 1.0
mmix 3 1.0, 0.0, 0.0, 0.0
mmix 4 1.0, 1.0, -1.0, -1.0
save
```

Modes

There are various modes that can be toggled on or off. Modes can be enabled/disabled by stick positions, auxillary receiver channels and other events such as failsafe detection.

ID	Short Name	Function
0	ARM	Enables motors and flight stabilisation
1	ANGLE	Legacy auto-level flight mode
2	HORIZON	Auto-level flight mode
4	ANTI GRAVITY	Prevents dips and rolls on fast throttle changes
5	MAG	Heading lock
6	HEADFREE	Head Free - When enabled yaw has no effect on pitch/roll inputs
7	HEADADJ	Heading Adjust - Sets a new yaw origin for HEADFREE mode
8	CAMSTAB	Camera Stabilisation
12	PASSTHRU	Pass roll, yaw, and pitch directly from rx to servos in airplane mix
13	BEEPERON	Enable beeping - useful for locating a crashed aircraft
15	LEDLOW	Switch off LED_STRIP output
17	CALIB	Start in-flight calibration
19	OSD	Enable/Disable On-Screen-Display (OSD)
20	TELEMETRY	Enable telemetry via switch
23	SERVO1	Servo 1
24	SERVO2	Servo 2
25	SERVO3	Servo 3
26	BLACKBOX	Enable BlackBox logging
27	FAILSAFE	Enter failsafe stage 2 manually
28	AIRMODE	Alternative mixer and additional PID logic for more stable copter
29	3D	Enable 3D mode
30	FPV ANGLE MIX	Apply yaw rotation relative to a FPV camera mounted at a preset angle
31	BLACKBOX ERASE	Erase the contents of the onboard flash log chip (takes > 30 s)
32	CAMERA CONTROL 1	Control function 1 of the onboard camera (if supported)
33	CAMERA CONTROL 2	Control function 2 of the onboard camera (if supported)
34	CAMERA CONTROL 3	Control function 3 of the onboard camera (if supported)
35	FLIP OVER AFTER CRASH	Reverse the motors to flip over an upside down craft after a crash (DShot required)
36	BOXPREARM	When arming, wait for this switch to be activated before actually arming
37	BEEP GPS SATELLITE COUNT	Use a number of beeps to indicate the number of GPS satellites found
39	VTX PIT MODE	Switch the VTX into pit mode (low output power, if supported)
40	USER1	User defined switch 1. Intended to be used to control an arbitrary output with PINIO
41	USER2	User defined switch 2. Intended to be used to control an arbitrary output with PINIO
		User defined switch 3. Intended to be used to control an arbitrary

42 USER3	output with PINIO
43 USER4	User defined switch 4. Intended to be used to control an arbitrary output with PINIO
44 PID AUDIO	Enable output of PID controller state as audio
45 PARALYZE	Permanently disable a crashed craft until it is power cycled
46 GPS RESCUE	Enable 'GPS Rescue' to return the craft to the location where it was last armed
47 ACRO TRAINER	Enable 'acro trainer' angle limiting in acro mode

Auto-leveled flight

The default flight mode does not stabilize the multicopter around the roll and the pitch axes. That is, the multicopter does not level on its own if you center the pitch and roll sticks on the radio. Rather, they work just like the yaw axis: the rate of rotation of each axis is controlled directly by the related stick on the radio, and by leaving them centered the flight controller will just try to keep the multicopter in whatever orientation it's in. This default mode is called "Rate" mode, also sometime called "Acro" (from "acrobatic") or "Manual" mode, and is active whenever no auto-leveled mode is enabled.

If your flight controller is equipped with a 3 axis accelerometer (very likely), then you can enable one of the two available auto leveled flight modes.

Mode details

Angle

In this auto-leveled mode the roll and pitch channels control the angle between the relevant axis and the vertical, achieving leveled flight just by leaving the sticks centered.

Horizon

This hybrid mode works exactly like the previous ANGLE mode with centered roll and pitch sticks (thus enabling auto-leveled flight), then gradually behaves more and more like the default RATE mode as the sticks are moved away from the center position.

Headfree

In this mode, the "head" of the multicopter is always pointing to the same direction as when the feature was activated. This means that when the multicopter rotates around the Z axis (yaw), the controls will always respond according the same "head" direction.

With this mode it is easier to control the multicopter, even fly it with the physical head towards you since the controls always respond the same. This is a friendly mode to new users of multicopters and can prevent losing the control when you don't know the head direction.

Airmode

In the standard mixer / mode, when the roll, pitch and yaw gets calculated and saturates a motor, all motors
will be reduced equally. When motor goes below minimum it gets clipped off.
Say you had your throttle just above minimum and tried to pull a quick roll - since two motors

can't go
any lower, you essentially get half the power (half of your PID gain).
If your inputs would asked for more than 100% difference between the high and low motors, the low motors
would get clipped, breaking the symmetry of the motor balance by unevenly reducing the gain.
Airmode will enable full PID correction during zero throttle and give you ability for nice zero throttle
gliding and actobatics. But also the cornering / turns will be much tighter now as there is always maximum
possible correction performed. Airmode can also be enabled to work at all times by always putting it on the
same switch like your arm switch or you can enable/disable it in air. Additional things and benefits: Airmode
will additionally fully enable Iterm at zero throttle. Note that there is still some protection on the ground
when throttle zeroed (below min\check) and roll/pitch sticks centered. This is a basic protection to limit
motors spooling up on the ground. Also the Iterm will be reset above 70% of stick input in acro mode to prevent
quick Iterm windups during finishes of rolls and flips, which will provide much cleaner and more natural stops
of flips and rolls what again opens the ability to have higher I gains for some.
Note that AIRMODE will also overrule motor stop function! It will basically also act as an idle up switch.

Auxiliary Configuration

Spare auxillary receiver channels can be used to enable/disable modes. Some modes can only be enabled this way.

Configure your transmitter so that switches or dials (potentiometers) send channel data on channels 5 and upwards (the first 4 channels are usually occupied by the throttle, aileron, rudder, and elevator channels).

e.g. You can configure a 3 position switch to send 1000 when the switch is low, 1500 when the switch is in the middle and 2000 when the switch is high.

Configure your tx/rx channel limits to use values between 1000 and 2000. The range used by mode ranges is fixed to 900 to 2100.

When a channel is within a specifed range the corresponding mode is enabled.

Use the GUI configuration tool to allow easy configuration when channel.

CLI

There is a CLI command, aux that allows auxillary configuration. It takes 5 arguments as follows:

- AUD range slot number (0 - 39)
- mode id (see mode list above)
- AUX channel index (AUX1 = 0, AUX2 = 1,... etc)
- low position, from 900 to 2100. Should be a multiple of 25.

- high position, from 900 to 2100. Should be a multiple of 25.

If the low and high position are the same then the values are ignored.

e.g.

Configure AUX range slot 0 to enable ARM when AUX1 is withing 1700 and 2100.

```
aux 0 0 0 1700 2100
```

You can display the AUX configuration by using the aux command with no arguments.

Oneshot

Oneshot allows faster communication between the Flight Controller and the ESCs that are present on your multirotor.

It does this in two ways:

1. Use a signal that varies between 125 µs and 250 µs (instead of the normal PWM timing of 1000µs to 2000µs)
2. Only send a 'shot' once per flight controller loop, and do this as soon as the flight controller has calculated the required speed of the motors.

Supported ESCs

FlyDuino KISS ESCs are able to use the Oneshot125 protocol out of the box. There is only one soldering needed.
BLHeli rev13.0 also supports Oneshot125 and will be automatically selected by the ESC without additional work.

Supported Boards

The Naze boards are supported, and have been flight tested in a number of configurations.

CC3D boards have been tested with a PPM receiver, however parallel PWM receivers might not work properly with this board.

Enabling Oneshot

To configure Oneshot, you must turn off any power to your ESCs.

It is a good idea at this stage to configure your ESC for oneshot mode (by soldering JP1 in the case of the KISS ESC).

Connect a USB cable to your board, and connect using the Chrome GUI app.

Go to the CLI tab, and type the following:

```
feature ONESHOT125
save
```

Then you can safely power up your ESCs again.

Configuration

The process for calibrating oneshot ESCs is the same as any other ESC.

1. Ensure that your ESCs are not powered up.
2. Connect to the board using a USB cable, and change to the motor test page.
3. Set the motor speed to maximum using the main slider.
4. Connect power to your ESCs. They will beep.
5. Click on the slider to bring the motor speed down to zero. The ESCs will beep again,

usually a couple of times.
6. Disconnect the power from your ESCs.
7. Re-connect power to your ESCs, and verify that moving the motor slider makes your motors spin up normally.

References

- FlyDuino (http://flyduino.net/)

Profiles

A profile is a set of configuration settings.

Currently three profiles are supported. The default profile is profile `0`.

Changing profiles

Profiles can be selected using a GUI or the following stick combinations:

Profile	Throttle	Yaw	Pitch	Roll
0	Down	Left	Middle	Left
1	Down	Left	Up	Middle
2	Down	Left	Middle	Right

The CLI `profile` command can also be used:

```
profile <index>
```

Rate Profiles

Cleanflight supports rate profiles in addition to regular profiles.

Rate profiles contain settings that adjust how your craft behaves based on control input.

Three rate profiles are supported.

Rate profiles can be selected while flying using the inflight adjustments feature.

Each normal profile has a setting called 'default_rate_profile`. When a profile is activated the corresponding rate profile is also activated.

Profile 0 has a default rate profile of 0.
Profile 1 has a default rate profile of 1.
Profile 2 has a default rate profile of 2.

The defaults are set this way so that it's simple to configure a profile and a rate profile at the same.

The current rate profile can be shown or set using the CLI `rateprofile` command:

```
rateprofile <index>
```

The values contained within a rate profile can be seen by using the CLI `dump rates` command.

e.g

```
# dump rates

# rateprofile
rateprofile 0

set rc_rate = 90
set rc_expo = 65
set thr_mid = 50
set thr_expo = 0
set roll_pitch_rate = 0
set yaw_rate = 0
set tpa_rate = 0
set tpa_breakpoint = 1500
```

RSSI

RSSI is a measurement of signal strength and is very handy so you know when your aircraft is going out of range or if it is suffering RF interference.

Some receivers have RSSI outputs. 3 types are supported.

1. RSSI via PPM channel
2. RSSI via Parallel PWM channel
3. RSSI via ADC with PPM RC that has an RSSI output - aka RSSI ADC

RSSI via PPM

Configure your receiver to output RSSI on a spare channel, then select the channel used via the CLI.

e.g. if you used channel 9 then you would set:

```
set rssi_channel = 9
```

Note: Some systems such as EZUHF invert the RSSI (0 = Full signal / 100 = Lost signal). To correct this problem you can invert the channel input so you will get a correct reading by using command:

```
set rssi_invert = ON
```

Default is set to "OFF" for normal operation (100 = Full signal / 0 = Lost signal).

RSSI via Parallel PWM channel

Connect the RSSI signal to any PWM input channel then set the RSSI channel as you would for RSSI via PPM

RSSI from Futaba S.Bus receiver

The S.Bus serial protocol includes detection of dropped frames. These may be monitored and reported as RSSI by using the following command.

```
set rssi_src_frame_errors = ON
```

Note that RSSI stands for Received Signal Strength Indicator; the detection of S.Bus dropped frames is really a signal quality, not strength indication. Consequently you may experience a more rapid drop in reported RSSI at the extremes of range when using this facility than when using RSSI reporting signal strength.

RSSI ADC

Connect the RSSI signal to the RC2/CH2 input. The signal must be between 0v and 3.3v.
Use inline resistors to lower voltage if required; inline smoothing capacitors may also help.
A simple PPM->RSSI conditioner can easily be made. See the PPM-RSSI conditioning.pdf for details.

Under CLI :

- enable using the RSSI_ADC feature : `feature RSSI_ADC`
- set the RSSI_SCALE parameter (between 1 and 255) to adjust RSSI level according to your configuration. The raw ADC value is divided by the value of this parameter.

Note: Some systems invert the RSSI (0 = Full signal / 100 = Lost signal). To correct this problem you can invert the input so you will get a correct reading by using command:

```
set rssi_invert = ON
```

RSSI_SCALE setup method:

- set rssi_scale = 100. The displayed percentage will then be the raw ADC value.
- turn on RX (close to board). RSSI value should vary a little.
- Update rssi_scale to the maximum RSSI value previously measured.

FrSky D4R-II and X8R supported.

The feature can not be used when RX_PARALLEL_PWM is enabled.

Receivers (RX)

A receiver is used to receive radio control signals from your transmitter and convert them into signals that the flight controller can understand.

There are 3 basic types of receivers:

1. Parallel PWM Receivers
2. PPM Receivers
3. Serial Receivers

As of 2016 the recommendation for new purchases is a Serial or PPM based receiver. Avoid Parallel PWM recievers (1 wire per channel). This is due to the amount of IO pins parallel PWM based receivers use. Some new FC's do not support parallel PWM.

Parallel PWM Receivers

8 channel support, 1 channel per input pin. On some platforms using parallel input will disable the use of serial ports
and SoftSerial making it hard to use telemetry or GPS features.

PPM Receivers

PPM is sometimes known as PPM SUM or CPPM.

12 channels via a single input pin, not as accurate or jitter free as methods that use serial communications, but readily available.

These receivers are reported working:

- FrSky D4R-II (http://www.frsky-rc.com/product/pro.php?pro_id=24)
- Graupner GR24 (http://www.graupner.de/en/products/33512/product.aspx)
- R615X Spektrum/JR DSM2/DSMX Compatible 6Ch 2.4GHz Receiver w/CPPM (http://www.hobbyking.com/hobbyking/store/__46632__OrangeRx_R615X_DSM2_DSMX_C
- FrSky D8R-XP 8ch telemetry receiver, or CPPM and RSSI enabled receiver (http://www.frsky-rc.com/product/pro.php?pro_id=21)
- FrSky X4R and FrSky X4RSB (http://www.frsky-rc.com/download/view.php?sort=&down=158&file=X4R-X4RSB) when flashed with CPPM firmware and bound with jumper between signal pins 2 and 3
- All FrSky S.Bus enabled devices when connected with S.Bus CPPM converter cable (http://www.frsky-rc.com/product/pro.php?pro_id=112). Without jumper this converter cable uses 21ms frame size (Channels 1-8). When jumper is in place, it uses 28ms frame and channels 1-12 are available
- FlySky/Turnigy FS-iA4B, FS-iA6B, FS-iA10 receivers all provide 8channels if the tx is sending them. (FS-i6 and FS-i10 transmitters). Use setting rx-setup/ppm to enable.

Serial Receivers

Spektrum

8 channels via serial currently supported.

These receivers are reported working:

Lemon Rx DSMX Compatible PPM 8-Channel Receiver + Lemon DSMX Compatible Satellite with Failsafe
http://www.lemon-rx.com/index.php?route=product/product&product_id=118

S.BUS

16 channels via serial currently supported. See below how to set up your transmitter.

- You probably need an inverter between the receiver output and the flight controller. However, some flight controllers have this built in (the main port on CC3D, for example), and doesn't need one.
- Some OpenLRS receivers produce a non-inverted SBUS signal. It is possible to switch SBUS inversion off using CLI command `set sbus_inversion = OFF` when using an F3 based flight controller.
- Softserial ports cannot be used with SBUS because it runs at too high of a bitrate (1Mbps). Refer to the chapter specific to your board to determine which port(s) may be used.
- You will need to configure the channel mapping in the GUI (Receiver tab) or CLI (`map` command). Note that channels above 8 are mapped "straight", with no remapping.

These receivers are reported working:

FrSky X4RSB 3/16ch Telemetry Receiver
http://www.frsky-rc.com/product/pro.php?pro_id=135

FrSky X8R 8/16ch Telemetry Receiver
http://www.frsky-rc.com/product/pro.php?pro_id=105

Futaba R2008SB 2.4GHz S-FHSS
http://www.futaba-rc.com/systems/futk8100-8j/

OpenTX S.BUS configuration

If using OpenTX set the transmitter module to D16 mode and ALSO select CH1-16 on the transmitter before binding to allow reception
of all 16 channels.

OpenTX 2.09, which is shipped on some Taranis X9D Plus transmitters, has a bug - [issue:1701 (https://github.com/opentx/opentx/issues/1701)](https://github.com/opentx/opentx/issues/1701).
The bug prevents use of all 16 channels. Upgrade to the latest OpenTX version to allow correct reception of all 16 channels,
without the fix you are limited to 8 channels regardless of the CH1-16/D16 settings.

SRXL (formerly XBUS)

(Serial Receiver Link Protocol)
SRXL is an open data transfer protocol which allows to transport control data from a rc receiver to another device like a flybarless system
by only using one single line. This protocol has been established by SRXL.org based on the idea to create a freely available and unified protocol

that manufacturers can easily implement to their receivers and devices that process receiver data. The protocol does not describe an exact definition of
how the data must be processed. It only describes a framework in which receiver data can be packed. Each manufacturer can have his own ID, which must be
attached to the beginning of each data set, so that the device using this data can correctly identify and process the payload of the dataset.

Supported receivers:

Multiplex:

All receivers with SRXL (also FLEXX receivers)

####Gaupner / SJ HOTT:
All receiver with SUMD support

Spektrum:

AR7700 / AR9020 receiver

JR:

JR X-BUS
Make sure to set your TX to use "MODE B" for XBUS in the TX menus!
See here for info on JR's XBUS protocol: http://www.jrpropo.com/english/propo/XBus/
These receivers are reported working:

XG14 14ch DMSS System w/RG731BX XBus Receiver
http://www.jramericas.com/233794/JRP00631/

Jeti:

Receivers with UDI output

XBUS MODE B RJ01

There exist a remote receiver made for small BNF-models like the Align T-Rex 150 helicopter. The code also supports using the Align DMSS RJ01 receiver directly with the cleanflight software.
To use this receiver you must power it with 3V from the hardware, and then connect the serial line as other serial RX receivers.
In order for this receiver to work, you need to specify the XBUS_MODE_B_RJ01 for serialrx_provider. Note that you need to set your radio mode for XBUS "MODE B" also for this receiver to work.
Receiver name: Align DMSS RJ01 (HER15001)

SUMD

16 channels via serial currently supported.

These receivers are reported working:

GR-24 receiver HoTT
http://www.graupner.de/en/products/33512/product.aspx

Graupner receiver GR-12SH+ HoTT
http://www.graupner.de/en/products/870ade17-ace8-427f-943b-657040579906/33565/product.aspx

SUMH

8 channels via serial currently supported.

SUMH is a legacy Graupner protocol. Graupner have issued a firmware updates for many recivers that lets them use SUMD instead.

IBUS

10 channels via serial currently supported.

IBUS is the FlySky digital serial protocol and is available with the FS-IA4B, FS-IA6B and FS-IA10 receivers. The Turnigy TGY-IA6B and TGY-IA10 are the same devices with a different label, therefore they also work.

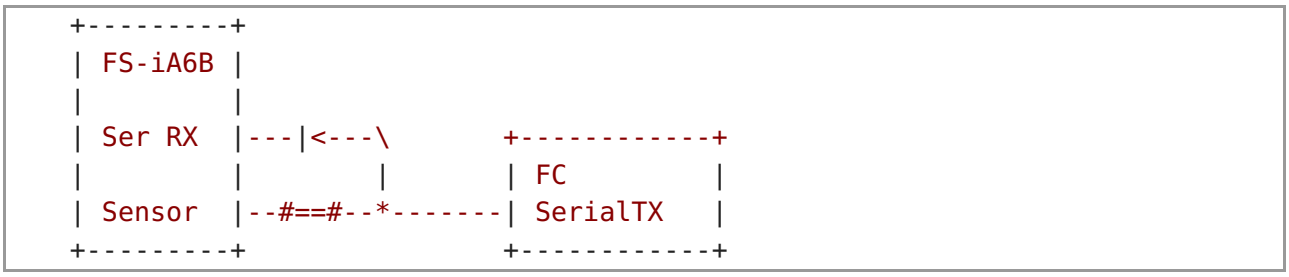
If you are using a 6ch tx such as the FS-I6 or TGY-I6 then you must flash a 10ch firmware on the tx to make use of these extra channels.

These receivers are reported working (all gives 10 channels serial):

- FlySky/Turnigy FS-iA4B 4-Channel Receiver (http://www.flysky-cn.com/products_detail/productId=46.html)
- FlySky/Turnigy FS-iA6B 6-Channel Receiver (http://www.flysky-cn.com/products_detail/&productId=51.html)
- FlySky/Turnigy FS-iA10 10-Channel Receiver (http://www.flysky-cn.com/products_detail/productId=53.html)
- FlySky/Turnigy FS-iA10B 10-Channel Receiver (http://www.flysky-cn.com/products_detail/productId=52.html)

Combine flysky ibus telemetry and serial rx on the same FC serial port

Connect Flysky FS-iA6B receiver like this:



Use a diode with cathode to receiver serial rx output (for example 1N4148), the anode is connected to the FC serial *TX* pin, and also via a resistor (10KOhm) to the receiver ibus sensor port.

Note (2018-07-27): In some cases, the value of the series resistor may be too large, and going down to 1K[ohm] may provide a good result.

Enable with cli:

```
serial 1 1088 115200 57600 115200 115200
feature RX_SERIAL
set serialrx_provider = IBUS
save
```

MultiWii serial protocol (MSP)

Allows you to use MSP commands as the RC input. Only 8 channel support to maintain compatibility with MSP.

Configuration

There are 3 features that control receiver mode:

```
RX_PPM
RX_SERIAL
RX_PARALLEL_PWM
RX_MSP
```

Only one receiver feature can be enabled at a time.

RX signal-loss detection

The software has signal loss detection which is always enabled. Signal loss detection is used for safety and failsafe reasons.

The `rx_min_usec` and `rx_max_usec` settings helps detect when your RX stops sending any data, enters failsafe mode or when the RX looses signal.

By default, when the signal loss is detected the FC will set pitch/roll/yaw to the value configured for `mid_rc`. The throttle will be set to the value configured for `rx_min_usec` or `mid_rc` if using 3D feature.

Signal loss can be detected when:

1. no rx data is received (due to radio reception, recevier configuration or cabling issues).
2. using Serial RX and receiver indicates failsafe condition.
3. using any of the first 4 stick channels do not have a value in the range specified by `rx_min_usec` and `rx_max_usec`.

RX loss configuration

The `rxfail` cli command is used to configure per-channel rx-loss behaviour.
You can use the `rxfail` command to change this behaviour.
A flight channel can either be AUTOMATIC or HOLD, an AUX channel can either be SET or HOLD.

- AUTOMATIC - Flight channels are set to safe values (low throttle, mid position for yaw/pitch/roll).
- HOLD - Channel holds the last value.
- SET - Channel is set to a specific configured value.

The default mode is AUTOMATIC for flight channels and HOLD for AUX channels.

The rxfail command can be used in conjunction with mode ranges to trigger various actions.

The `rxfail` command takes 2 or 3 arguments.

- Index of channel (See below)
- Mode ('a' = AUTOMATIC, 'h' = HOLD, 's' = SET)
- A value to use when in SET mode.

Channels are always specified in the same order, regardless of your channel mapping.

- Roll is 0
- Pitch is 1
- Yaw is 2
- Throttle is 3.
- Aux channels are 4 onwards.

Examples:

To make Throttle channel have an automatic value when RX loss is detected:

```
rxfail 3 a
```

To make AUX4 have a value of 2000 when RX loss is detected:

```
rxfail 7 s 2000
```

To make AUX8 hold it's value when RX loss is detected:

```
rxfail 11 h
```

WARNING: Always make sure you test the behavior is as expected after configuring rxfail settings!

`rx_min_usec`

The lowest channel value considered valid. e.g. PWM/PPM pulse length

`rx_max_usec`

The highest channel value considered valid. e.g. PWM/PPM pulse length

Serial RX

See the Serial chapter for some some RX configuration examples.

To setup spectrum on the Naze32 or clones in the GUI:

1. Start on the "Ports" tab make sure that UART2 has serial RX. If not set the checkbox, save and reboot.
2. Move to the "Configuration" page and in the upper lefthand corner choose Serial RX as the receiver type.
3. Below that choose the type of serial receiver that you are using. Save and reboot.

Using CLI:
For Serial RX enable RX_SERIAL and set the `serialrx_provider` CLI setting as follows.

Serial RX Provider	Value
SPEKTRUM1024	0

SPEKTRUM2048	1
SBUS	2
SUMD	3
SUMH	4
XBUS_MODE_B	5
XBUS_MODE_B_RJ01	6
IBUS	7

PPM/PWM input filtering.

Hardware input filtering can be enabled if you are experiencing interference on the signal sent via your PWM/PPM RX.

Use the `input_filtering_mode` CLI setting to select a mode.

Value	Meaning
OFF	Disabled
ON	Enabled

Receiver configuration.

FrSky D4R-II

Set the RX for 'No Pulses'. Turn OFF TX and RX, Turn ON RX. Press and release F/S button on RX. Turn off RX.

Graupner GR-24 PWM

Set failsafe on the throttle channel in the receiver settings (via transmitter menu) to a value below `rx_min_usec` using channel mode FAILSAFE.
This is the prefered way, since this is *much faster* detected by the FC then a channel that sends no pulses (OFF).

NOTE:
One or more control channels may be set to OFF to signal a failsafe condition to the FC, all other channels *must* be set to either HOLD or OFF.
Do **NOT USE** the mode indicated with FAILSAFE instead, as this combination is NOT handled correctly by the FC.

Receiver Channel Range Configuration.

The channels defined in CleanFlight are as follows:

Channel number	Channel name
0	Roll
1	Pitch
2	Yaw
3	Throttle

If you have a transmitter/receiver, that output a non-standard pulse range (i.e. 1070-1930 as some Spektrum receivers)
you could use rx channel range configuration to map actual range of your transmitter to 1000-2000 as expected by Cleanflight.

The low and high value of a channel range are often referred to as 'End-points'. e.g. 'End-point adjustments / EPA'.

All attempts should be made to configure your transmitter/receiver to use the range 1000-2000 *before* using this feature
as you will have less preceise control if it is used.

To do this you should figure out what range your transmitter outputs and use these values for rx range configuration.
You can do this in a few simple steps:

If you have used rc range configuration previously you should reset it to prevent it from altering rc input. Do so
by entering the following command in CLI:

```
rxrange reset
save
```

Now reboot your FC, connect the configurator, go to the `Receiver` tab move sticks on your transmitter and note min and
max values of first 4 channels. Take caution as you can accidentally arm your craft. Best way is to move one channel at
a time.

Go to CLI and set the min and max values with the following command:

```
rxrange <channel_number> <min> <max>
```

For example, if you have the range 1070-1930 for the first channel you should use `rxrange 0 1070 1930` in
the CLI. Be sure to enter the `save` command to save the settings.

After configuring channel ranges use the sub-trim on your transmitter to set the middle point of pitch, roll, yaw and throttle.

You can also use rxrange to reverse the direction of an input channel, e.g. `rxrange 0 2000 1000`.

Safety

As many can attest, multirotors and RC models in general can be very dangerous, particularly on the test bench. Here are some simple golden rules to save you a trip to the local ER:

- **NEVER** arm your model with propellers fitted unless you intend to fly!
- **Always** remove your propellers if you are setting up for the first time, flashing firmware, or if in any doubt.

Before Installing

Please consult the Cli (Cli.md), Controls (Controls.md), Failsafe (Failsafe.md) and Modes (Modes.md)
pages for further important information.

You are highly advised to use the Receiver tab in the CleanFlight Configurator, making sure your Rx channel
values are centered at 1500 (1520 for Futaba RC) with minimum & maximums of 1000 and 2000 (respectively)
are reached when controls are operated. Failure to configure these ranges properly can create problems, such as inability to arm (because you can't reach the endpoints) or immediate activation of
failsafe (Failsafe.md).

You may have to adjust your channel endpoints and trims/sub-trims on your RC transmitter to achieve the
expected range of 1000 to 2000.

The referenced values for each channel have marked impact on the operation of the flight controller and the
different flight modes.

Props Spinning When Armed

With the default configuration, when the controller is armed, the propellers *WILL* begin spinning at low speed.
We recommend keeping this setting as it provides a good visual indication the craft is armed.

If you wish to change this behavior, see the MOTOR_STOP feature in the Configurator and relevant documentation pages.
Enabling this feature will stop the props from spinning when armed.

Serial

Cleanflight has enhanced serial port flexibility but configuration is slightly more complex as a result.

Cleanflight has the concept of a function (MSP, GPS, Serial RX, etc) and a port (VCP, UARTx, SoftSerial x).
Not all functions can be used on all ports due to hardware pin mapping, conflicting features, hardware, and software
constraints.

Serial port types

- USB Virtual Com Port (VCP) - USB pins on a USB port connected directly to the processor without requiring
  a dedicated USB to UART adapter. VCP does not 'use' a physical UART port.
- UART - A pair of dedicated hardware transmit and receive pins with signal detection and generation done in hardware.
- SoftSerial - A pair of hardware transmit and receive pins with signal detection and generation done in software.

UART is the most efficient in terms of CPU usage.
SoftSerial is the least efficient and slowest, SoftSerial should only be used for low-bandwidth usages, such as telemetry transmission.

UART ports are sometimes exposed via on-board USB to UART converters, such as the CP2102 as found on the Naze and Flip32 boards.
If the flight controller does not have an on-board USB to UART converter and doesn't support VCP then an external USB to UART board is required.
These are sometimes referred to as FTDI boards. FTDI is just a common manufacturer of a chip (the FT232RL) used on USB to UART boards.

When selecting a USB to UART converter choose one that has DTR exposed as well as a selector for 3.3v and 5v since they are more useful.

Examples:

- FT232RL FTDI USB To TTL Serial Converter Adapter (http://www.banggood.com/FT232RL-FTDI-USB-To-TTL-Serial-Converter-Adapter-Module-For-Arduino-p-917226.html)
- USB To TTL / COM Converter Module buildin-in CP2102 (http://www.banggood.com/Wholesale-USB-To-TTL-Or-COM-Converter-Module-Buildin-in-CP2102-New-p-27989.html)

Both SoftSerial and UART ports can be connected to your computer via USB to UART converter boards.

Serial Configuration

Serial port configuration is best done via the configurator.

Configure serial ports first, then enable/disable features that use the ports. To configure SoftSerial ports the SOFTSERIAL feature must be also be enabled.

Constraints

If the configuration is invalid the serial port configuration will reset to its defaults and features may be disabled.

- There must always be a port available to use for MSP/CLI.
- There is a maximum of 2 MSP ports.
- To use a port for a function, the function's corresponding feature must be also be enabled.
  e.g. after configuring a port for GPS enable the GPS feature.
- If SoftSerial is used, then all SoftSerial ports must use the same baudrate.
- Softserial is limited to 19200 baud.
- All telemetry systems except MSP will ignore any attempts to override the baudrate.
- MSP/CLI can be shared with EITHER Blackbox OR telemetry. In shared mode blackbox or telemetry will be output only when armed.
- Smartport telemetry cannot be shared with MSP.
- No other serial port sharing combinations are valid.
- You can use as many different telemetry systems as you like at the same time.
- You can only use each telemetry system once. e.g. FrSky telemetry cannot be used on two port, but MSP Telemetry + FrSky on different ports is fine.

Configuration via CLI

You can use the CLI for configuration but the commands are reserved for developers and advanced users.

The `serial` CLI command takes 6 arguments.

1. Identifier (see serialPortIdentifier_e in the source)
2. Function bitmask (see serialPortFunction_e in the source)
3. MSP baud rate
4. GPS baud rate
5. Telemetry baud rate (auto baud allowed)
6. Blackbox baud rate

Baud Rates

The allowable baud rates are as follows:

Identifier	Baud rate
0	Auto
1	9600
2	19200
3	38400
4	57600
5	115200
6	230400
7	250000

Passthrough

Cleanflight can enter a special passthrough mode whereby it passes serial data through to a device connected to a UART/SoftSerial port. This is useful to change the configuration of a Cleanflight peripheral such as an OSD, bluetooth dongle, serial RX etc.

To initiate passthrough mode, use the CLI command `serialpassthrough` This command takes four arguments.

```
serialpassthrough <id> [baud] [mode] [DTR PINIO]
```

ID is the internal identifier of the serial port from Cleanflight source code (see serialPortIdentifier_e in the source). For instance UART1-UART4 are 0-3 and SoftSerial1/SoftSerial2 are 30/31 respectively. Baud is the desired baud rate, and mode is a combination of the keywords rx and tx (rxtx is full duplex). The baud and mode parameters can be used to override the configured values for the specified port. `DTR PINIO` identifies the PINIO resource which is optionally connected to a DTR line of the attached device.

For example. If you have your MWOSD connected to UART 2, you could enable communicaton to this device using the following command. This command does not specify the baud rate or mode, using the one configured for the port (see above).

```
serialpassthrough 1
```

If a baud rate is not specified, or is set to 0, then `serialpassthrough` supports changing of the baud rate over USB. This allows tools such as the MWOSD GUI to dynamically set the baud rate to, for example 57600 for reflashing the MWOSD firmware and then 115200 for adjusting settings without having to powercycle your flight control board between the two.

To use a tool such as the MWOSD GUI, it is necessary to disconnect or exit Cleanflight configurator.

To exit serial passthrough mode, power cycle your flight control board.

In order to reflash an Arduino based device such as a MWOSD via `serialpassthrough` if is necessary to connect the DTR line in addition to the RX and TX serial lines. The DTR is used as a reset line to invoke the bootloader. The DTR line may be connected to any GPIO pin on the flight control board. This pin must then be associated with a PINIO resource, the instance of which is then passed to the serialpassthrough command. The DTR line associated with any given UART may be set using the CLI command `resource` specifying it as a PINIO resource.

For example, the following configuration for an OpenPilot Revolution shows the UART6 serial port to be configured with TX on pin C06, RX on pin C07 and a DTR connection using PINIO on pin C08.

```
resource SERIAL_TX 1 A09
resource SERIAL_TX 3 B10
resource SERIAL_TX 4 A00
resource SERIAL_TX 6 C06
resource SERIAL_RX 1 A10
resource SERIAL_RX 3 B11
resource SERIAL_RX 6 C07

resource PINIO 1 C08
```

To assign the DTR line to another pin use the following command.

```
resource PINIO 1 c05
```

To disassociate DTR from a pin use the following command.

```
resource PINIO 1 none
```

Having configured a PINIO resource assocaited with a DTR line as per the above example, connection to an MWOSD attached to an Openpilot Revolution could be achieved using the following command.

```
serialpassthrough 5 0 rxtx 1
```

This will connect using UART 6, with the baud rate set over USB, full duplex, and with DTR driven on PINIO resource 1.

A (desirable) side effect of configuring the DTR line to be associated with a PINIO resource, is that when the FC is reset, the attached Arduino device will also be reset.

Note that if DTR is left configured on a port being used with a standard build of MWOSD firmware, the display will break-up when the flight controller is reset. This is because, by default, the MWOSD does not correctly handle resets from DTR. There are two solutions to this:

1. Assign the DTR pin using the resource command above prior to reflashing MWOSD, and then dissasociate DTR from the pin.
2. Rebuild MWOSD with MAX_SOFTRESET defined. The MWOSD will then be reset correctly every time the flight controller is reset.

Sonar

A sonar sensor can be used to measure altitude for use with BARO and SONAR altitude hold modes.

The sonar sensor is used instead of the pressure sensor (barometer) at low altitudes (less than about 3.5 meters above the ground).
The sonar sensor is only used when the aircraft inclination angle (attitude) is small (less than 22.5 degrees).

Hardware

Currently the main supported sensor is the HCSR04 sensor.
The Parallax 28015 single-wire sonar can also be used by connecting 1k resistors to the Trigger and Echo pins, and the other end of the resistors shorted together and to the Sonar module.

Connections

Naze/Flip32+

Mode	Trigger	Echo	Inline 1k resistors
Parallel PWM/ADC current sensor	PB8 / Motor 5	PB9 / Motor 6	NO (5v tolerant)
PPM/Serial RX	PB0 / RC7	PB1 / RC8	YES (3.3v input)

Constraints

Current meter cannot be used in conjunction with Parallel PWM and Sonar.

CC3D

Trigger	Echo	Inline 1k resistors
PB5 / RC4	PB0 / RC5	YES (3.3v input)

Constraints

Sonar cannot be used in conjuction with SoftSerial or Parallel PWM.

SPRacingF3

Trigger	Echo	Inline 1k resistors
PB0 / RC7	PB1 / RC8	YES (3.3v input)

Constraints

Sonar cannot be used in conjuction with SoftSerial2 or Parallel PWM.

Telemetry

Telemetry allows you to know what is happening on your aircraft while you are flying it. Among other things you can receive battery voltages and GPS positions on your transmitter.

Telemetry can be either always on, or enabled when armed. If a serial port for telemetry is shared with other functionality then telemetry will only be enabled when armed on that port.

Telemetry is enabled using the 'TELEMETRY` feature.

```
feature TELEMETRY
```

Multiple telemetry providers are currently supported, FrSky, Graupner
HoTT V4, SmartPort (S.Port), LightTelemetry (LTM) and Ibus

All telemetry systems use serial ports, configure serial ports to use the telemetry system required.

FrSky telemetry

FrSky telemetry is transmit only and just requires a single connection from the TX pin of a serial port to the RX pin on an FrSky telemetry receiver.

FrSky telemetry signals are inverted. To connect a cleanflight capable board to an FrSKy receiver you have some options.

1. A hardware inverter - Built in to some flight controllers.
2. Use software serial and enable frsky_inversion.
3. Use a flight controller that has software configurable hardware inversion (e.g. STM32F30x).

For 1, just connect your inverter to a usart or software serial port.

For 2 and 3 use the CLI command as follows:

```
set tlm_inversion = ON
```

Available sensors

The following sensors are transmitted :

Vspd : vertical speed, unit is cm/s.

Hdg : heading, North is 0°, South is 180°.

AccX,Y,Z : accelerometers values.

Tmp1 : baro temp if available, gyro otherwise.

RPM : if armed : throttle value, battery capacity otherwise. (Blade number needs to be set to 12 in Taranis).

VFAS : actual vbat value (see VFAS precision section bellow).

Curr : actual current comsuption, in amp.

Fuel : if capacity set :remaining battery percentage, mah drawn otherwise.

GPS : GPS coordinates.

Alt : barometer based altitude, init level is zero.

Date : time since powered.

GSpd : current speed, calculated by GPS.

GAlt : GPS altitude, sea level is zero.

Tmp2 : number of sats. Every second, a number > 100 is sent to represent GPS signal quality.

Cels : average cell value, vbat divided by cell number.

> Cleanflight will send Cels (FLVSS Individual Cell Voltages Telemetry), disable the setting to use actual FLVSS sensor with:
>
> ```
> set telemetry_send_cells = OFF
> ```
>
> Note: cell voltage values are an assumed reputation of the cell voltage based on the packs voltage. Actual cell voltage may differ.
>
> To view individual cells or more importantly to get lowest cell (all cells are the sum of vbat, so each cell is the same in this case):
> See [OpenTX 2.1 & FrSky FLVSS Individual Cell Voltages (http://openrcforums.com/forum/viewtopic.php?t=7266)](http://openrcforums.com/forum/viewtopic.php?t=7266).
> Add a new sensor, to display the lowest cell voltage set it up like this:
>
> - Type: Calculated
> - Formula: Cell
> - Cell Sensor: Cels *(pack total voltage, sum of all cells)*
> - Cell Index: Lowest

Precision setting for VFAS

Cleanflight can send VFAS (FrSky Ampere Sensor Voltage) in two ways:

```
set frsky_vfas_precision  = 0
```

This is default setting which supports VFAS resolution of 0.2 volts and is supported on all FrSky hardware.

```
set frsky_vfas_precision  = 1
```

This is new setting which supports VFAS resolution of 0.1 volts and is only supported by OpenTX radios (this method uses custom ID 0x39).

HoTT telemetry

Only Electric Air Modules and GPS Modules are emulated.

Use the latest Graupner firmware for your transmitter and receiver.

Older HoTT transmitters required the EAM and GPS modules to be enabled in the telemetry menu of the transmitter. (e.g. on MX-20)

You can connect HoTT-Telemetry in two ways:

Old way:

Serial ports use two wires but HoTT uses a single wire so some electronics are required so that the signals don't get mixed up. The TX and RX pins of
a serial port should be connected using a diode and a single wire to the T port on a HoTT receiver.

Connect as follows:

- HoTT TX/RX T -> Serial RX (connect directly)
- HoTT TX/RX T -> Diode -(|)- > Serial TX (connect via diode)

The diode should be arranged to allow the data signals to flow the right way

```
-(  |)- == Diode, | indicates cathode marker.
```

1N4148 diodes have been tested and work with the GR-24.

When using the diode disable `tlm_halfduplex`, go to CLI and type `set tlm_halfduplex = 0FF`, don't forget a `save` afterwards.

New way:

You can use a single connection, connect HoTT RX/TX only to serial TX, leave serial RX open and make sure `tlm_halfduplex` is ON.

As noticed by Skrebber the GR-12 (and probably GR-16/24, too) are based on a PIC 24FJ64GA-002, which has 5V tolerant digital pins.

Note: The SoftSerial ports may not be 5V tolerant on your board. Verify if you require a 5v/3.3v level shifters.

LightTelemetry (LTM)

LTM is a lightweight streaming telemetry protocol supported by a
number of OSDs, ground stations and antenna trackers.

The Cleanflight implementation of LTM implements the following frames:

- G-FRAME: GPS information (lat, long, ground speed, altitude, sat info)
- A-FRAME: Attitude (pitch, roll, heading)
- S-FRAME: Status (voltage, current+, RSSI, airspeed+, status). Item suffixed '+' not implemented in Cleanflight.
- O-FRAME: Origin (home position, lat, long, altitude, fix)

In addition, in the inav (navigation-rewrite) fork:

- N-FRAME: Navigation information (GPS mode, Nav mode, Nav action, Waypoint number, Nav Error, Nav Flags).

LTM is transmit only, and can work at any supported baud rate. It is designed to operate over 2400 baud (9600 in Cleanflight) and does not benefit from higher rates. It is thus usable on soft serial.

More information about the fields, encoding and enumerations may be found at
https://github.com/stronnag/mwptools/blob/master/docs/ltm-definition.txt

MAVLink telemetry

MAVLink is a very lightweight, header-only message marshalling library for micro air vehicles. Cleanflight supports MAVLink for compatibility with ground stations, OSDs and antenna trackers built
for PX4, PIXHAWK, APM and Parrot AR.Drone platforms.

MAVLink implementation in Cleanflight is transmit-only and usable on low baud rates and can be used over soft serial.

SmartPort (S.Port)

Smartport is a telemetry system used by newer FrSky transmitters and receivers such as the Taranis/XJR and X8R, X6R and X4R(SB).

More information about the implementation can be found here:
https://github.com/frank26080115/cleanflight/wiki/Using-Smart-Port

Available sensors

The following sensors are transmitted :

A4 : average cell value. Warning : unlike FLVSS sensors, you do not get actual lowest value of a cell, but an average : (total lipo voltage) / (number of cells)

Alt : barometer based altitude, init level is zero.

Vspd : vertical speed, unit is cm/s.

Hdg : heading, North is 0°, South is 180°.

AccX,Y,Z : accelerometers values.

Tmp1 : actual flight mode, sent as 4 digits. Number is sent as (1)1234. Please ignore the leading 1, it is just there to ensure the number as always 5 digits (the 1 + 4 digits of actual data) the numbers are aditives (for example, if first digit after the leading 1 is 6, it means GPS Home and Headfree are both active) :

1. 1 is GPS Hold, 2 is GPS Home, 4 is Headfree
2. 1 is mag enabled, 2 is baro enabled, 4 is sonar enabled
3. 1 is angle, 2 is horizon, 4 is passthrough
4. 1 is ok to arm, 2 is arming is prevented, 4 is armed

Tmp2 : GPS lock status, Number is sent as 1234, the numbers are aditives :

1. 1 is GPS Fix, 2 is GPS Home fix
2. not used
3. not used
4. number of sats

VFAS : actual vbat value.

GAlt : GPS altitude, sea level is zero.

GSpd : current speed, calculated by GPS.

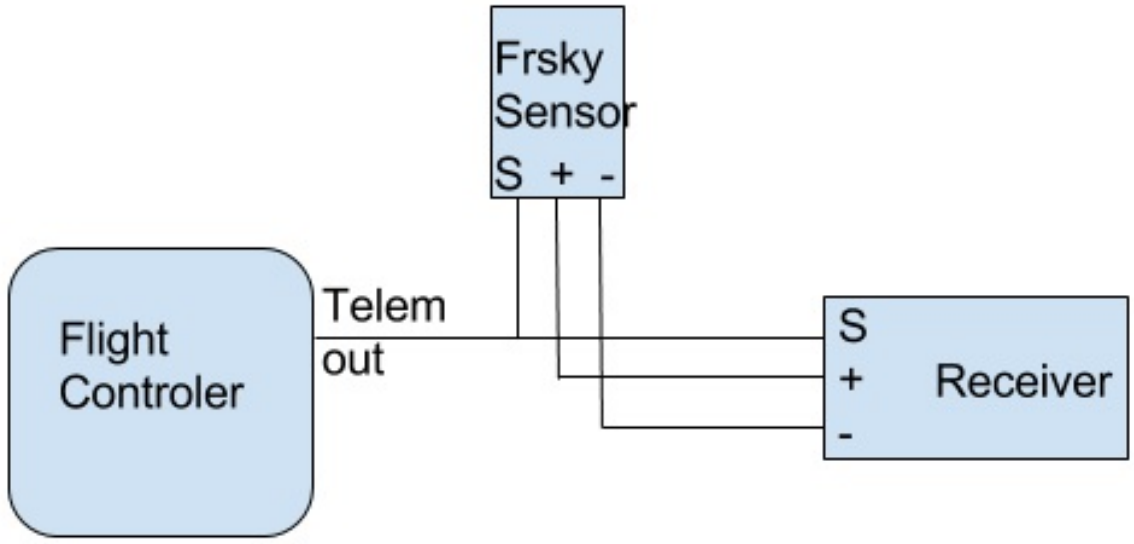
GPS : GPS coordinates.

Cels : average cell value, vbat divided by cell number.

> Cleanflight will send Cels (FLVSS Individual Cell Voltages Telemetry), disable the setting to use actual FLVSS sensor with:
>
> ```
> set telemetry_send_cells = OFF
> ```
>
> Note: cell voltage values are an assumed reputation of the cell voltage based on the packs voltage. Actual cell voltage may differ. It is recommeded that you chain the flight controllers telemetry with a real Frsky FLVSS s.port sensor.
>
> To view individual cells or more importantly to get lowest cell (all cells are the sum of vbat, so each cell is the same in this case):
> See [OpenTX 2.1 & FrSky FLVSS Individual Cell Voltages (http://openrcforums.com/forum/viewtopic.php?t=7266)](http://openrcforums.com/forum/viewtopic.php?t=7266).
> Add a new sensor, to display the lowest cell voltage set it up like this:
>
> - Type: Calculated
> - Formula: Cell
> - Cell Sensor: Cels *(pack total voltage, sum of all cells)*
> - Cell Index: Lowest

Integrate Cleanflight telemetry with FrSky Smartport sensors

While Cleanflight telemetry brings a lot of valuable data to the radio, there are additional sensors, like Lipo cells sensor FLVSS, that can be a great addition for many aircrafts. Smartport sensors are designed to be daisy chained, and CF telemetry is no exception to that. To add an external sensor, just connect the "S" port of the FC and sensor(s) together, and ensure the sensor(s) are getting connected to GND and VCC either from the controler or the receiver



SmartPort on F3 targets with hardware UART

Smartport devices can be connected directly to STM32F3 boards such as the SPRacingF3 and Sparky, with a single straight through cable without the need for any hardware modifications on the FC or the receiver. Connect the TX PIN of the UART to the Smartport signal pin.

For Smartport on F3 based boards, enable the telemetry inversion setting.

```
set tlm_inversion = ON
```

SmartPort on F1 and F3 targets with SoftSerial

Since F1 targets like Naze32 or Flip32 are not equipped with hardware inverters, SoftSerial might be simpler to use.

1. Enable SoftSerial `feature SOFTSERIAL`
2. In Configurator assign *Telemetry* > *Smartport* > *Auto* to SoftSerial port of your choice
3. Enable Telemetry `feature TELEMETRY`
4. Confirm telemetry inversion `set tlm_inversion = ON`
5. You have to bridge TX and RX lines of SoftSerial and connect them together to S.Port signal line in receiver

Notes:

- This has been tested with Flip32 and SPracingF3 boards and FrSky X8R and X4R receivers
- To discover all sensors board has to be armed, and when GPS is connected, it needs to have a proper 3D fix. When not armed, values like ***Vfas*** or GPS coordinates may not sent.

Ibus telemetry

Ibus telemetry requires a single connection from the TX pin of a bidirectional serial port to the Ibus sens pin on an FlySky telemetry receiver. (tested with fs-iA6B receiver, iA10 should work)

It shares 1 line for both TX and RX, the rx pin cannot be used for other serial port stuff.
It runs at a fixed baud rate of 115200.

It should be possible to daisy chain multiple sensors with ibus. This is implemented but not tested because i don't have one of the sensors to test with, the FC shall always be the last sensor in the chain.

It is possible to combine serial rx and ibus telemetry on the same uart pin on the flight controller, see Rx (Rx.md).

Configuration

Ibus telemetry can be enabled in the firmware at build time using defines in target.h. It is enabled by default in those targets that have space left.

```
#define TELEMETRY
#define TELEMETRY_IBUS
```

CLI command to enable:

```
serial 1 1024 115200 57600 115200 115200
```

CLI setting to determine if the voltage reported is Vbatt or calculated average cell voltage

```
set ibus_report_cell_voltage=[ON/OFF]
```

Available sensors

The following sensors are transmitted :

Tmp1 : baro temp if available, gyro otherwise.

RPM : throttle value

Vbatt : configurable battery voltage or the average cell value, vbat divided by number of cells.

RX hardware

These receivers are reported to work with i-bus telemetry:

- FlySky/Turnigy FS-iA6B 6-Channel Receiver (http://www.flysky-cn.com/products_detail/&productId=51.html)
- FlySky/Turnigy FS-iA10B 10-Channel Receiver (http://www.flysky-cn.com/products_detail/productId=52.html)

Note that the FlySky/Turnigy FS-iA4B 4-Channel Receiver (http://www.flysky-cn.com/products_detail/productId=46.html) seems to work but has a bug that might lose the binding, DO NOT FLY the FS-iA4B!

Transponder

Cleanflight supports the generation of race transponder signals on compatiable F3 and F4 targets.

IR led connections is target specific. Please consult the reference manual for your FC.

Cleanflight currently supports 3 tranponder protocol providers (as of 2.10).

iLap Provider

Links:

Web (http://www.rclapcounter.com/)
Contact (cs@rclapcounter.com)

Description:

iLap is a commercial system that uses 6 byte transponder codes and a 460kHz carrier.

Transponder codes are entered in the CF Configurator Transponder tab as 12 hex digits.

Codes are theoretical unique. Codes are obtained for iLap or come with some flight controllers.

<img src="Screenshots/Provider%20iLap.png" width="800">

ArcTimer Provider

Links:

Web (http://www.arcitimer.com)
Contact (info@arcitimer.com)

Description:

Arctimer is a commercial system that uses 9 byte transponder codes and a 42kHz carrier.

There are only 9 unique Arctimer codes. Codes are pick from a list on CF Configurator Transponder tab.

<img src="Screenshots/Provider%20ArcTimer.png" width="800">

EasyRaceLapTimer (ERLT) Provider

Links:

Web (http://www.easyracelaptimer.com/)
Facebook (https://www.facebook.com/groups/1015588161838713/)
RCGroups (https://www.rcgroups.com/forums/showthread.php?2538917-EasyRaceLapTimer-open-source-and-open-hardware-FPV-racing-lap-time-tracking-system)
GitHub (https://github.com/polyvision/EasyRaceLapTimer)

Description

EasyRaceLapTimer is a open source system that uses 6bit transponder codes and a 38kHz carrier.

There are 64 unique ERLT codes. Codes are pick from a list on CF Configurator Transponder tab.

<img src="Screenshots/Provider%20ERLT.png" width="800">

VTX

Cleanflight supports control of VTX modules.

VTX Systems

Current support includes

1. RTC6705 directly connected to the CPU (maybe via a PCB board interconnect, e.g. SPRACINGF3NEO)
2. IRC Tramp
3. TBS Smart Audio

VTX Button

If your FC has a button, excluding a BOOT buttons, then it can be used for VTX control.

Some boards like the SPRacingF3NEO have both a VTX module and a button.
Other boards like the SPRacingF3MINI have multiple buttons.

VTX Button usage

While the VTX button is held the STATUS 2 LED will flash N times per second indicating the action that will be taken when the button is released. The flashing starts as soon as the button is held. e.g. You press the button, count flashes and then release as appropriate.

Duration	Function	Flashes
25ms to 1s	Cycle Channel	4
1s to 3s	Cycle Band	3
3s to 5s	Cycle Power and RF Power	2
5s or more	Save FC settings	1

Example to cycle VTX power

```
0 seconds       | 1 second      | 2 seconds     | 3 seconds     | 4 seconds
| 5 seconds     | 6 seconds or more |
[-HOLD BUTTON-----------------------------------|-RELEASE BUTTON-NOW----------
-|-RELEASED TO LATE TO CHANGE POWER -|
| 4 Flashes     | 3 flashes     | 3 flashes     | 2 flashes     | 2 flashes
| 1 flash       | 1 flash           |
```

The VTX button works with ALL VTX systems including onboard RTC6705, Tramp and SmartAudio.

If the VTX can be turned off then POWER 0 will turn off the VTX and POWER 1 will set the VTX into it's lowest power output.
If the VTX cannot be turned off then POWER 0 will set the VTX into it's lowest power output.