# Documentation for the Test-Flight-Logging Software (tflog) and Libraries (tflog-libs) for PX4 Quadrotor Systems and the NPoint Tracking Software

# 1 The PX4 Quadrotor System: An open-source development and research platform for MAVs (= Micro-Air-Vehicles)

## *1.1 The Flight Management Unit (PX4FMU)*

http://pixhawk.org/modules/px4fmu

### 1.1.1 The Processing Unit

- 168 MHz / 252 MIPS Cortex-M4F
- Hardware floating point unit (FPU)
- POSIX-compatible RTOS, NuttX OS
- SIMD extensions
- 192KB SRAM / 1024 KB Flash

### 1.1.2 The Sensors

http://pixhawk.org/users/apps/sensors

– 3D accelerometer and gyroscope
    - MPU-6000
    - 500 Hz update rate

– 3D gyroscope
    - L3GD20
    - 500 Hz update rate

- 3D magnetometer
  - HMC5883L
  - 100 Hz update rate

- barometric pressure sensor
  - MS5611
  - 100 Hz

- voltage measurement
  - on-chip ADC
  - for battery status
  - 83 Hz

### 1.1.3 Attitude Estimator

- Extended Kalman Filter (EKF)
- inputs (250 Hz update rate):
  - accelerometer
  - gyroscope
  - magnetometer

- estimates
  - Euler angles: Roll, Pitch, Yaw (RPY)
  - angular speeds

- update rate
  - update rate >= 150 Hz
  - depends on system load

### 1.1.4 Attitude Control

http://pixhawk.org/users/apps/multirotor_att_control

- update rate: 250 Hz
- manual control mode is used

### 1.1.5 Weight and Payload

| | | |
|---|---|---|
| – basic PX4 Drone without batteries: | 210g | |
| – basic PX4 Drone with batteries: | 380g | |
| – maximum payload to add: | 210g | (total 590g) |

## 1.2 *The ARDrone Quadrotor Air Frame and the Motor Driver Board (PX4ARIO)*

http://pixhawk.org/modules/px4ioar

- the following Parrot ARDrone spare parts are used to build the quadrotor system

  1x Parrot Central Cross (Mfr Part# PF070008AA)
  4x Parrot Motor Set for AR.Drone 2.0 (Mfr Part# PF070040AA)
  1x Parrot Propellers for AR.Drone 2.0 (Mfr Part# PF070045AA)
  1x Parrot Gears & Shafts for AR.Drone 2.0 (Mfr Part# PF070047AA)

- NOTE: battery power cables with TAMIYA connectors don't come along with the PX4 packages

## 1.3 *Assemble a new PX4 Drone*

- assembling the airframe
  - mount the motors into the parrot central cross
  - mount the propellers using the gears and shafts

- before the motors can be used with the PX4ARIO driver board, they must be flashed twice with the original Parrot Motor firmware

- this can be done by connecting the motors to an AR.Drone 1 mainboard and by booting the OS twice

- disconnect parrot central cross from the AR.Drone 1 mainboard
- mount PX4FMU on PX4ARIO
- connect central cross cables to PX4ARIO
- mount parrot central cross onto the PX4ARIO board
- find power cables, build TAMIYA female/male connector (10mm x 6mm x 22mm)
- solder cables to PX4ARIO board

# 2 Prepare a PC as flight base platform

## 2.1 *PX4 Drone Firmware*

### 2.1.1 Resolve Dependencies

https://pixhawk.ethz.ch/px4/dev/toolchain_installation_lin

– install the needed software packages

```
sudo apt-get update
sudo apt-get upgrade

sudo apt-get install python-serial python-argparse openocd \
flex bison libncurses5-dev autoconf texinfo build-essential \
libftdi-dev libtool zlib1g-dev genromfs ia32-libs git-core wget
```

– install toolchain for cross-compiling the firmware
– get latest ARM cross-compiler from launchpad website

https://launchpad.net/gcc-arm-embedded

– use the version “gcc-arm-none-eabi-4_8-2013q4” (which is known to work)
– add the toolchain to the path (restart is needed afterwards)

```
exportline="export PATH=$HOME/gcc-arm-none-eabi-4_8-2013q4/bin:\$PATH"
```

– temporarily add toolchain to path of current console

```
if grep -Fxq "$exportline" ~/.profile; then echo nothing to do ; else
echo $exportline >> ~/.profile; fi
. ~/.profile
```

– add the current user to the dialout group (needed for accessing PX4FMU and for flashing firmware)

```
sudo usermod -a -G dialout $USER
```

### 2.1.2 Get Modified Firmware Source Code (mandatory)

https://pixhawk.ethz.ch/px4/dev/nuttx/building_and_flashing_console

– get the modified firmware (there are two options)
  - use the Firmware which comes along with this documentation

    commit dec695f6a175eb61461e347ceea8a42e3971f203

    - get the latest Firmware from my git repository

- get the firmware from the git repository
  - clone git repository

    ```
    git clone git@bitbucket.org:fbergner/nst_msc_px4fmu.git
    ```

  - rename the folder of the repository to „Firmware“

- enter the „Firmware“ directory and copy „NuttX“ folder into that directory
- again we have two options
  - clone a fresh NuttX git repository into the Firmware folder

    ```
    git clone https://github.com/PX4/NuttX.git
    ```

  - use the “NuttX” folder which comes along with this documentation

    commit c4a201b471a1fc25874e9e5ce1ec73a4e85aeeef

### 2.1.3 Get the latest unmodified Firmware Source Code (optional)

```
git clone https://github.com/PX4/Firmware
cd Firmware
git clone https://github.com/PX4/NuttX.git
```

### 2.1.4 Build Firmware

https://pixhawk.ethz.ch/px4/dev/nuttx/building_and_flashing_console

- make sure that you are in the “Firmware” Folder
- compiling sequence at first compile time or when changes on the NuttX System have been made (e.g. new apps were added to nsh (NuttX shell), configuration file was changed)

  ```
  make distclean
  make archives
  make px4fmu-v1_default
  ```

- compiling sequence when only apps have been changed

  ```
  make clean
  make px4fmu-v1_default
  ```

- almost always using only the following instruction is sufficient

  ```
  make px4fmu-v1_default
  ```

### 2.1.5 Flash Firmware

- connect PX4FMU via USB to the PC
- start the flashing process

```
make upload px4fmu-v1_default
```

- when bootloader is not found right away: Press the reset button (small button on the left side next to the USB connector)

## 2.2 *QGroundcontrol Software*

### 2.2.1 Resolve Dependencies

- install all the needed packages

```
sudo apt-get install phonon libqt4-dev libphonon-dev libphonon4 phonon-
backend-gstreamer qtcreator libsdl1.2-dev libflite1 flite1-dev build-
essential libopenscenegraph-dev
```

- install qt-sdk: Download it and follow instructions

  https://qt-project.org/downloads

### 2.2.2 Get Source Code (optional)

- download source code

```
git clone git://github.com/mavlink/qgroundcontrol.git
cd qgroundcontrol
git submodule init
git submodule update
```

### 2.2.3 Build “libxbee” Library (mandatory)

- the library source files can be found in the folder

  qgroundcontrol/libs/thirdParty/libxbee

- alternatively: use the libxbee source files which come along with this documentation

- read the “README” file

– compile and install the library

```
make install
```

### 2.2.4 Build Binary From Source Code (optional)

– make sure that the library “libxbee” was successfully installed
– open the qgroundcontrol.pro project file with qtcreator
– start the building process
– building the program from source code might take a while (~ 15 min)
– find the built binary in the folder

qgroundcontrol/release

– a binary compiled for ubuntu 64bit comes along with this documentation

commit d2e513c558a66c5efd4da11a972759a2ed3fdc11

## *2.3 The tflog Software*

– clone the following git repositories into a common folder

```
git clone git@bitbucket.org:fbergner/tflog.git
git clone git@bitbucket.org:fbergner/tflog-libs.git
```

– these two folders also come along with this documentation

- tflog:

  commit fc16c76365812a542779e0786bd8a498deb36e30

- tflog-libs:

  commit 3a50db3daa884d2bc9186450620cf2bd7862835f

– the common folder should now contain the following folders

tflog
tflog-libs

– build the “tflog” libraries

```
cd tflog-libs
make clean && make all
```

– build the “tflog” program

```
cd tflog
make clean && make all
```

– the binary can be found in the “tflog” folder

# 3 Prepare the PX4 Drone and the PC programs for the first flight

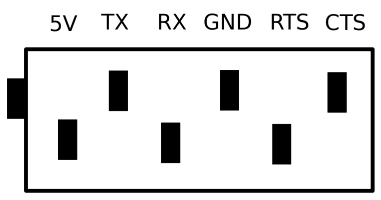
## 3.1 *Prepare SD-Card for PX4 Drone*

https://pixhawk.ethz.ch/px4/users/apps/auto_starting_apps

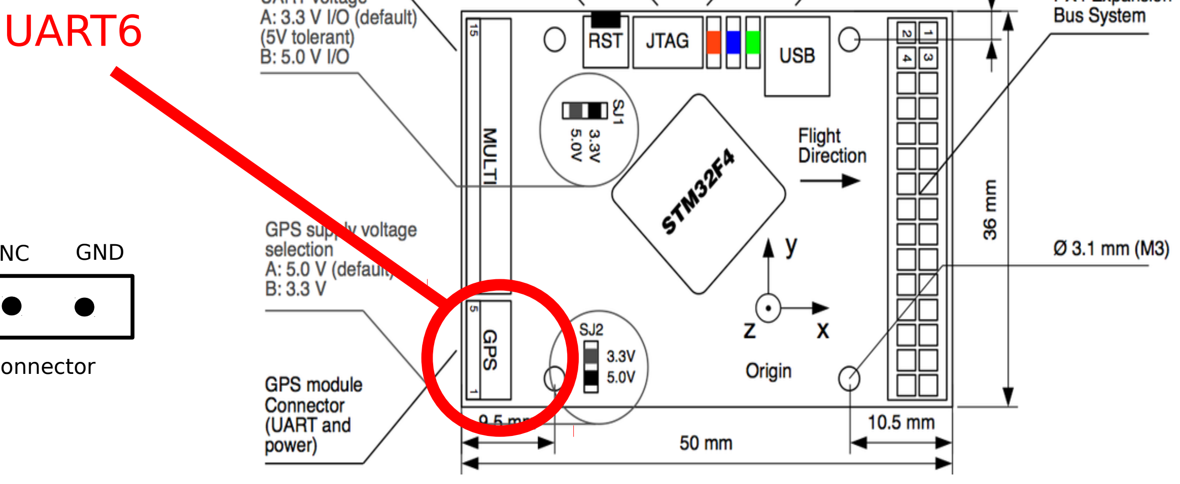
- format SD card with a FAT file system
- start and use disk utility on Linux
- create a "etc" folder on the SD card
- create a startup script "rc" and place it into the folder "etc"
- use a working startup script which comes along with this documentation (there are two versions)
  - one which enables motor arming for standard use and for flying the PX4 Drone (v0.0.4fa)
  - one which disables motor arming for testing purposes; flying the PX4 Drone is not possible (v0.0.4f)

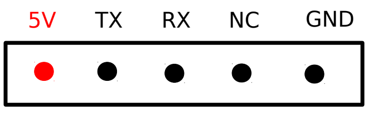
## 3.2 *Connect a NST WiFi Module to the PX4 Drone*

- use a 5 terminal Molex cable
- cut the cable in half
- crimp the cable into a 6 terminal male micro-match connector
- only connect the TX, RX signals and GND
- cross the TX, RX signals



Front side of male micro-match connector


UART6
Peripherals, Computer, Radio, and motors
ARM Mini JTAG (10 pos)
Reset Button
Status Leds Green: Power on Blue: Activity Amber: Error
USB Micro-B Virt. COM port USB powered
3.0 mm
PX4 Expansion Bus System
Telemetry UART voltage A: 3.3 V I/O (default) (5V tolerant) B: 5.0 V I/O
RST
JTAG
USB
MULTI
15
SJ1
3.3V
5.0V
STM32F4
Flight Direction
36 mm
Ø 3.1 mm (M3)
GPS supply voltage selection A: 5.0 V (default) B: 3.3 V
GPS
SJ2
y
z
x
Origin
GPS module Connector (UART and power)
9.5 mm
50 mm
10.5 mm



Front side of molex connector

- connect the created cable to UART6 of the PX4 Drone
- connect the cable to the WiFi module
- supply the WiFi module with an independent power source (1 LiPo cell, 3.7V)
- the 5V power supply of the PX4 Drone is not powerful enough to support the WiFi module

### 3.3 *Configure the tflog program*

- the configuration of the tflog program is set with the “config.cfg” file
- adjust the settings to your specific setup
- change the argument of

  DEFAULT_MAV_CONN_TCP_SERVER_ADDR

  to the ip address of the WiFi module you use

- change the argument of

  DEFAULT_MAV_CONN_TCP_SERVER_PORT

  to the port used by the WiFi module

- change the argument of

  DEFAULT_JS_DEVICE

  to the path of your joystick device

### 3.4 *Calibrate the Joystick*

- install the program “jstest-gtk”

  ```
  sudo apt-get install jstest-gtk
  ```

- run the program

  ```
  jstest-gtk
  ```

- select the right joystick by double clicking it
- follow the calibration instructions and quit the program
- NOTE: Whenever the joystick is disconnected or you restart your system you have to redo the joystick calibration process

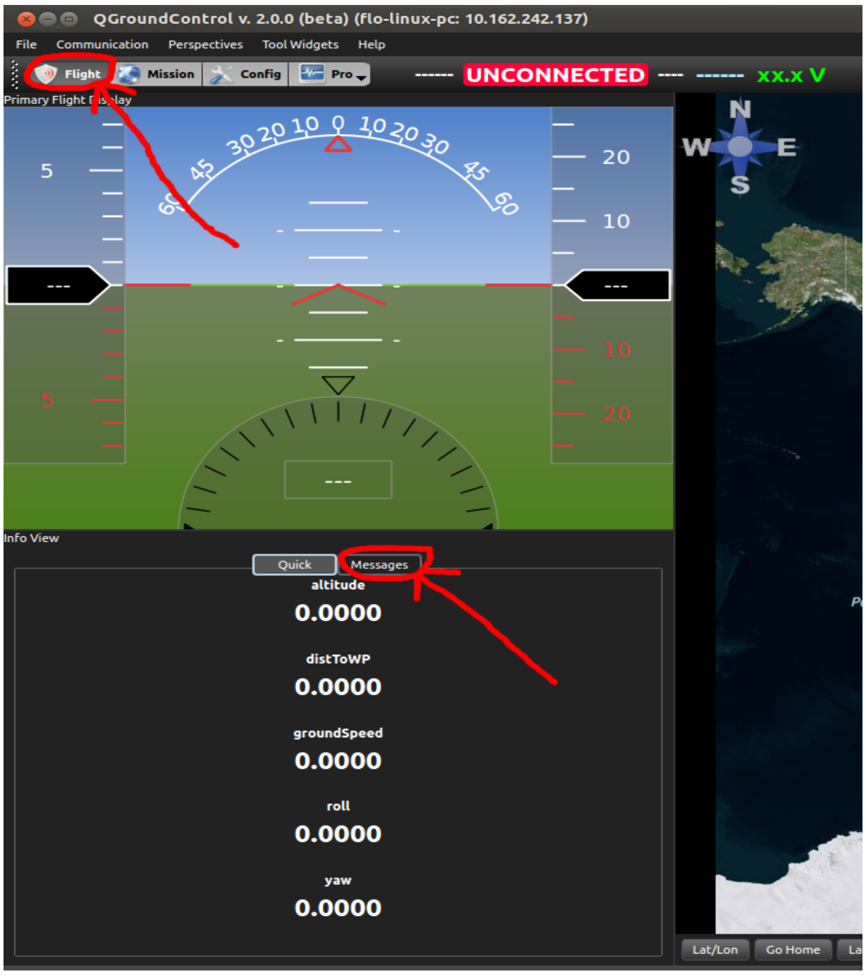
# 4 Establish a connection between PX4 Drone, tflog program and QGroundcontrol

## *4.1 The startup Process*

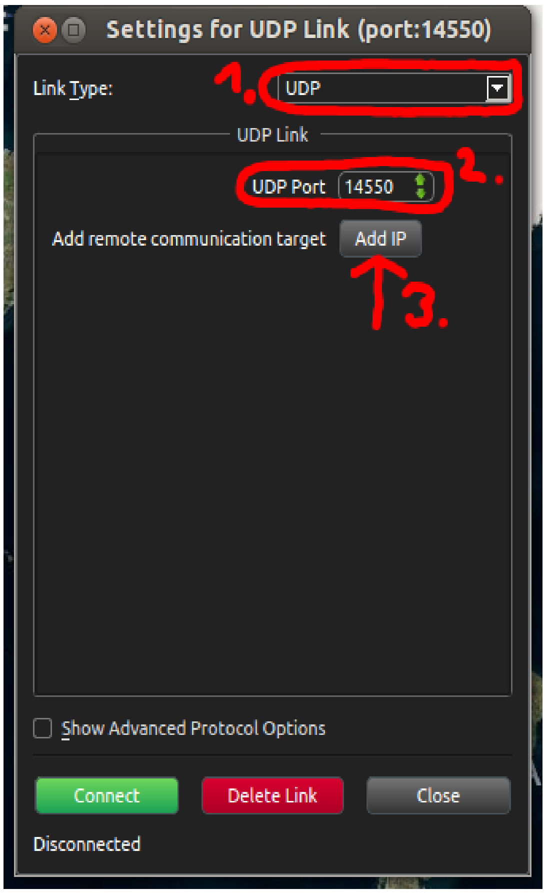
1. Supply the WiFi module with power and wait until you can ping it with the PC
2. Power up the PX4 Drone
3. Start the tflog program; the program should be able to establish a connection to the WiFi module and it should start receiving and sending MAVLINK packages (the tflog program can be quit by using “CTRL+C”)
4. Start QGroundcontrol and establish a connection to the UDP server of the tflog program
5. If the connection is good: QGroundcontrol recognizes the PX4 Drone and displays its status (e.g. battery voltage)

## *4.2 QGroundcontrol: Establish a connection to an UDP server*

1. Start QGroundcontrol
2. Change to the “Flight” perspective, switch to the “Messages” tab and click the big green “Connect Link” button



3. The settings dialog opens


Settings for UDP Link (port:14550)
Link Type:
UDP
1.
UDP Link
UDP Port
14550
2.
Add remote communication target
Add IP
3.
Show Advanced Protocol Options
Connect
Delete Link
Close
Disconnected

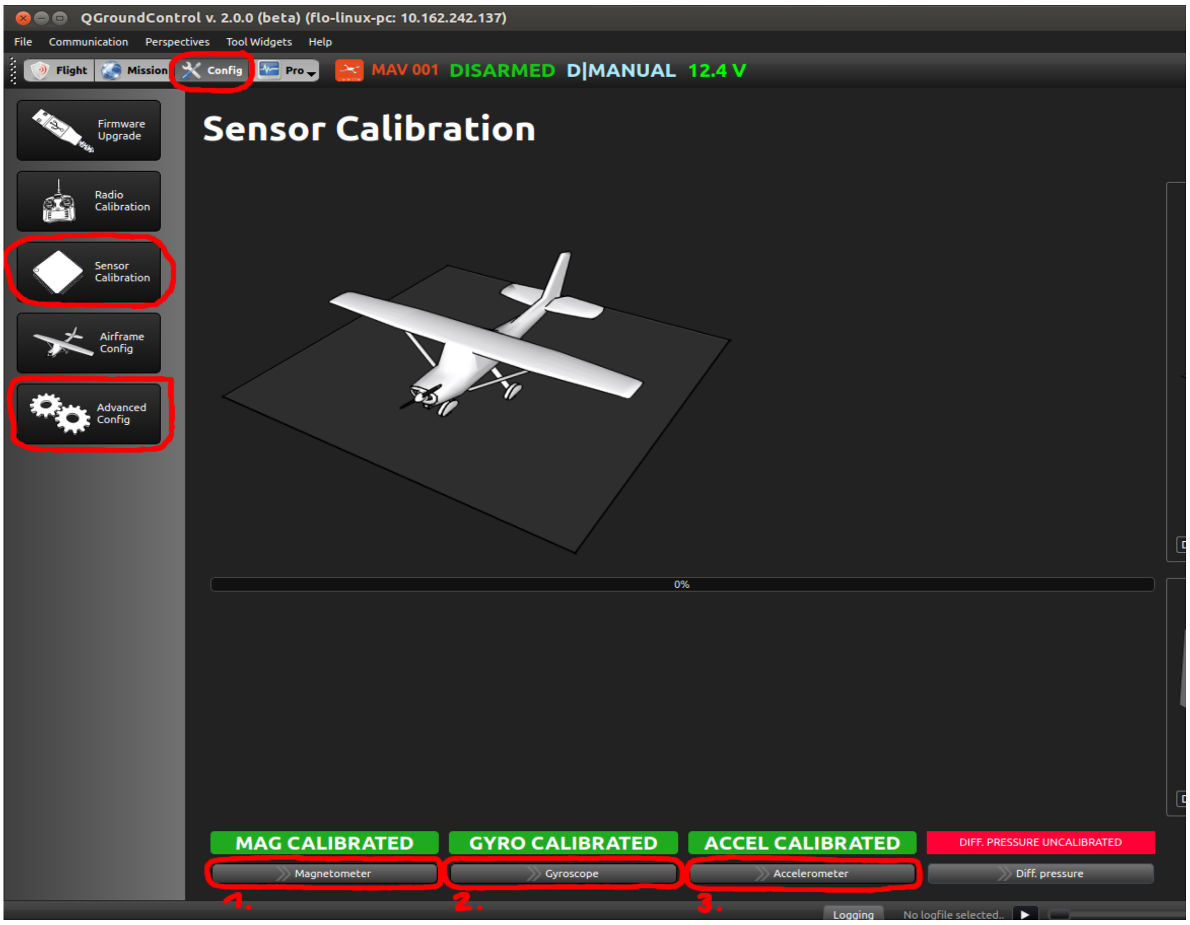
   1. Select UDP as link type
   2. Set the UDP port to 5555
   3. Click the Add IP button; the UDP server settings dialog opens
   4. Change the server setting to “localhost:5556”
   5. Close the server settings dialog by pressing the “OK” button
   6. Close the settings dialog by pressing the “Connect” button

4. QGroundcontrol now tries to establish a connection to the specified UDP server

# 5 Calibrate the PX4 Drone

## 5.1 *Sensor calibration (mandatory for new, uncalibrated systems)*

1. Change to the “Config” perspective
2. Press the “Sensor Calibration” button
3. Start the magnetometer calibration process by pressing the “Mangnetometer” button
   - Follow the calibration instructions which are displayed above the buttons
   - 1. step: the PX4 Drone must rest on flat ground
   - 2. step: the PX4 Drone must be taken in hand and the figure of a laying 8 must be “flown” (the figure is displayed) until the calibration completes
4. Start the gyroscope calibration process by pressing the “Gyroscope” button
   - Follow the instructions
   - the PX4 Drone must rest on flat ground and must not be moved until the calibration process completes
5. Start the Accelerometer calibration process by pressing the “Accelerometer” button
   - Follow the instructions and move to the displayed orientations
   - the PX4 Drone must again be taken in hand and orientated as displayed
   - the you must hold on orientation until the calibration process moves to the next step

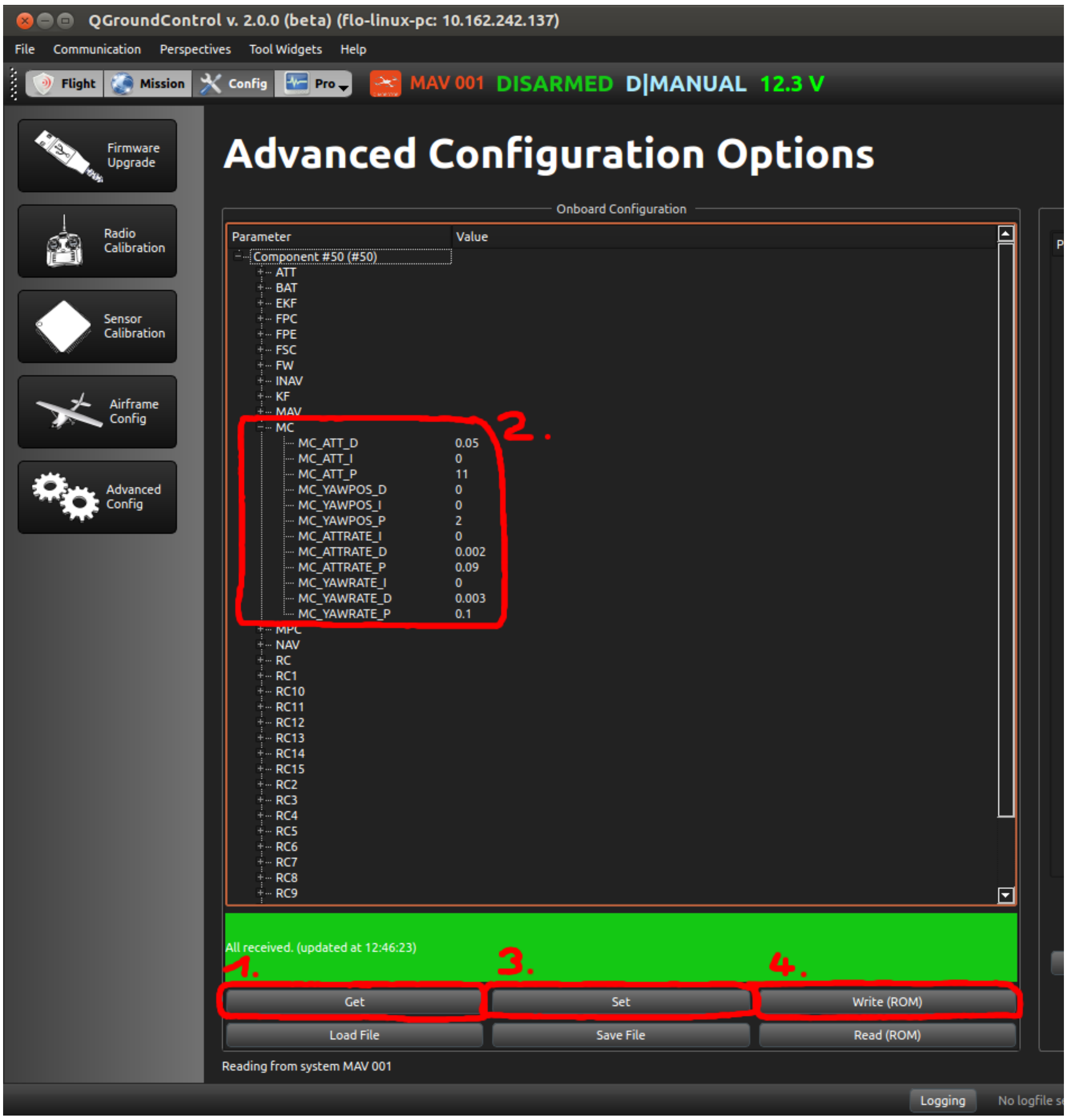


## 5.2 *PID controller Tuning (optional)*

- the PID controller parameters are already set to a working set of values (the parameter file which has previously been copied to the SD Card contains this set of values)
- if readjustments are needed or if you want to find a completely new set of values the following descriptions might help

### 5.2.1 Adjust some PID controller values

1. Change to the “Config” perspective
2. Click the “Advanced Config” button



| Parameter | Value |
|---|---|
| Component #50 (#50) | |
| ATT | |
| BAT | |
| EKF | |
| FPC | |
| FPE | |
| FSC | |
| FW | |
| INAV | |
| KF | |
| MAV | |
| MC | |
| MC_ATT_D | 0.05 |
| MC_ATT_I | 0 |
| MC_ATT_P | 11 |
| MC_YAWPOS_D | 0 |
| MC_YAWPOS_I | 0 |
| MC_YAWPOS_P | 2 |
| MC_ATTRATE_I | 0 |
| MC_ATTRATE_D | 0.002 |
| MC_ATTRATE_P | 0.09 |
| MC_YAWRATE_I | 0 |
| MC_YAWRATE_D | 0.003 |
| MC_YAWRATE_P | 0.1 |
| MPC | |
| NAV | |
| RC | |
| RC1 | |
| RC10 | |
| RC11 | |
| RC12 | |
| RC13 | |
| RC14 | |
| RC15 | |
| RC2 | |
| RC3 | |
| RC4 | |
| RC5 | |
| RC6 | |
| RC7 | |
| RC8 | |
| RC9 | |

All received. (updated at 12:46:23)

Reading from system MAV 001

1. Press the “Get” button to get a set of all the parameters
2. The PID controller parameters are in the MC section; they can be changed by double-clicking them; then new values can be entered
3. Press the “Set” button to transfer the new parameters to the PX4 Drone; the values are stored to volatile memory; the parameters don't survive a power-off power-on sequence
4. Press the “Write (ROM)” button to store parameters to non-volatile memory

#### 5.2.2 Find a new set of usable PID values

https://pixhawk.ethz.ch/px4/users/multirotor_pid_tuning

– have a look at file “px4-pid-calib.txt” and follow the instructions

# 6 Fly the PX4 Drone

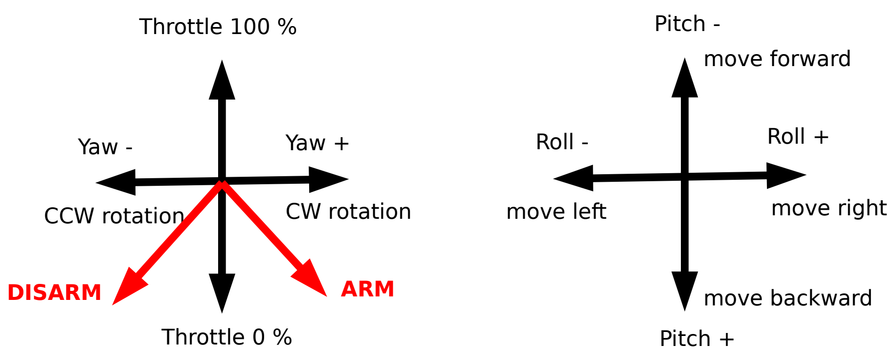
## *6.1 Startup Sequence*

1. Make sure that you have a calibrated joystick
2. Make sure that you have inserted a SD Card with a startup script which allows you to arm and disarm the PX4 Drone (NOTE: there a two different kind of startup scripts)
3. Place the PX4 Drone on flat ground
4. Connect the battery to the WiFi module
5. Switch on the PX4 Drone
6. Start the tflog program
7. Optional: Start the QGroundcontrol software to get a better overview about the current status of the PX4 Drone
8. Arm the PX4 Drone by moving the left stick of the joystick to the bottom-right corner for about 4 secs until the propellers of the PX4 start to rotate in free-running mode
9. Fly the PX4 Drone !

## *6.2 Shutdown Sequence*

1. Land the PX4 Drone
2. Disarm the PX4 Drone by moving the left stick of the joystick to the bottom-left corner for about 4 secs
3. Quit the tflog program by pressing “CTRL+C”
4. Disconnect the battery from the WiFi module
5. Switch off PX4 Drone
6. Disconnect battery from PX4 Drone

### 6.3 *The Joystick axes*



# 7 Use the NPoint Tracker in combination with the tflog software

### 7.1 *NPoint Tracker Startup Sequence*

- NOTE: the tracker startup sequence must be applied BEFORE the tflog software is started; after the tflog software has detected tracker packages the settings of the tracker software (e.g. name of rigid bodies) must NOT be changed;

1. Start the tracker program “Motive”
2. Select the latest calibration setting
3. Select markers and bind them to a rigid body; the tflog software only logs rigid bodies; for each rigid body an independent logging file is created
4. Select the streaming menu and check the broadcast checkbox; now dynamic tracker packets are sent via the UDP multicast protocol

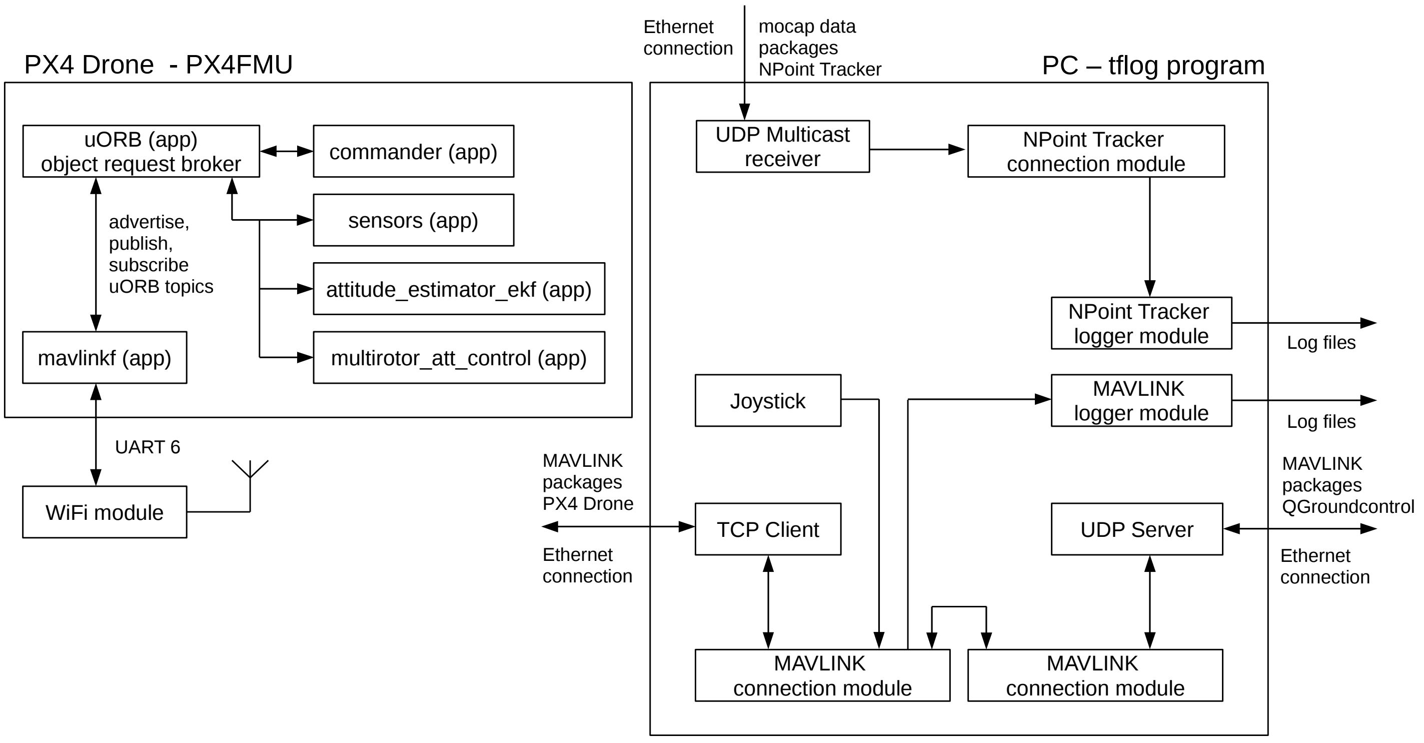
# 8 The MAVLINK protocol and the tflog Software

## 8.1 The MAVLINK protocol

https://pixhawk.ethz.ch/mavlink/

– a lightweight, multi-platform, standard communication protocol for communications between a MAV (Micro-Air-Vehicle) and a ground station (like QGroundcontrol)
– this protocol is supported by the PX4 Drone and used
  • to send commands from the ground station to the PX4 Drone
  • to receive at the ground station status messages, measurements and parameters from the PX4 Drone

## 8.2 The Tasks of the tflog Program


PX4 Drone - PX4FMU
uORB (app) object request broker
commander (app)
sensors (app)
attitude_estimator_ekf (app)
multirotor_att_control (app)
advertise, publish, subscribe uORB topics
mavlinkf (app)
UART 6
WiFi module
Ethernet connection
mocap data packages NPoint Tracker
PC – tflog program
UDP Multicast receiver
NPoint Tracker connection module
NPoint Tracker logger module
Log files
Joystick
MAVLINK logger module
Log files
MAVLINK packages PX4 Drone
Ethernet connection
TCP Client
UDP Server
MAVLINK packages QGroundcontrol
Ethernet connection
MAVLINK connection module
MAVLINK connection module

– the tflog software has the following tasks
– establish a TCP connection to the WiFi module of the PX4 Drone
– the WiFi module is the TCP server
– the tflog program connects as TCP Client to that server
– the tflog program implements an UDP server and the QGroundcontrol software connects to that server as UDP client
– the UDP server of the tflog program only accepts one client; all other attempts to connect to the server are ignored
– MAVLINK packages are sent and received over the TCP and the UDP link

- received packages are parsed and if the parsing was successful then the packets are forwarded to the other connection
- so the tflog program acts as message bridge and connection translator (UDP ↔ TCP)
- the tflog program interprets joystick events and transforms them to manual control values
- these manual control values are sent to the PX4 Drone using the manual_control packet of the MAVLINK protocol
- all important MAVLINK messages which come from the PX4 Drone are logged into text files; there is one log file for each logged MAVLINK message
- the tflog program also logs tracker data from a NPoint tracking system
- all functionality of the tflog program is bundled in libraries (tflog-libs) so that functionality can easily be used in other programs
- the tflog program supports the following operation modes
  - log PX4 Drone data only, no tracking system is connected
  - log tracking data only, no PX4 Drone is connected
  - log PX4 Drone data and tracking data at the same time
  - connecting a joystick is optional; the program also works without a connected joystick

# 9 Adjustments applied to the original PX4FMU firmware

- modified firmware bases on the original firmware with

  ```
  commit f1fece2bb6fe4d40128f3f17b92c073d50cce982
  ```

- the most recent commit of the modified firmware when this documentation was made is

  ```
  commit dec695f6a175eb61461e347ceea8a42e3971f203
  ```

- the modifications:
  - simplified build process so that only the needed binaries are compiled and thus the build process speeds up; modified “Firmware/Makefile”

  - removed “tone_alarm” from boot-up process by modifying “Firmware/ROMFS/px4fmu_common/init.d/rcS”

  - added additional modules to the firmware; modified “Firmware/makefiles/config_px4fmu-v1_default.mk ” so that these modules are built and added to the nsh (the NuttX shell)

  - modified the NuttX configuration file; rearranged the serial console and reconfigured the UARTs for our use (baudrate, DMA, etc.); NOTE: a serial console works only on a UART without a DMA; high baudrates are only supported if the DMA is enabled; modified “Firmware/nuttx-configs/px4fmu-v1/nsh/defconfig”

# 10 More Details on the PX4FMU Firmware

- more details can be found in the “px4-doc” documentation folder of this documentation
- the “px4-doc” folder contains multiple text files with brief answers to various questions related to the firmware which might still not be answered by this document