HoverGames Drone Hands-On Workshop

Leutrim Mustafa & Iain Galloway

Drone Program, Commercial Open-Source
Small Autonomous Vehicle for Robotic Drones and Rovers

June 2019 | Session # AMF-AUT-T3665A/B

SECURE CONNECTIONS
FOR A SMARTER WORLD

Agenda

- HoverGames Challenge
- HoverGames Drone
- Communication Methods
- Flight Controller
- Documentation
- QGroundcontrol
- PX4 Software Stack
- Example Code

Plane

Flying Wing

VTOL
Pusher Prop

Long Flight Duration

VTOL Transitioning Wing

Airship

Submarine

Delivery Rover

Boat

Security

Lawnmower or Agricultural?

HoverGames Drone

Reference Design

- Complete low cost ‘hobby’ drone platform, but really an open design robot
- 500mm size big enough for easy experimentation
- Complete system to test new components such as motor controllers with UAVCAN or secure authentication of battery
- Reuse of components for ground Rovers

Hovergames Challenges

- Use a complete autonomous vehicle development platform an infrastructure
- Coding challenges with larger societal impact theme
- New technologies are introduced to the system for continual challenges
- Complete modular NXP development platform with flight management unit (FMU) using Kinetis K66 Arm-Cortex M4 MCU

The First **HoverGames** Challenge

FIGHT FIRE WITH FLYERS

At any given time around the world, an active wildfire is causing tremendous devastation. And it is not just forest fires that cause immeasurable damage. Structure fires in urban areas also pose a significant problem. In every case, our first responders were put on the front lines.

How can you use drone technology to help firefighters?

Join today at www.hovergames.com

Thank you to our partners

auterion

SEGGER

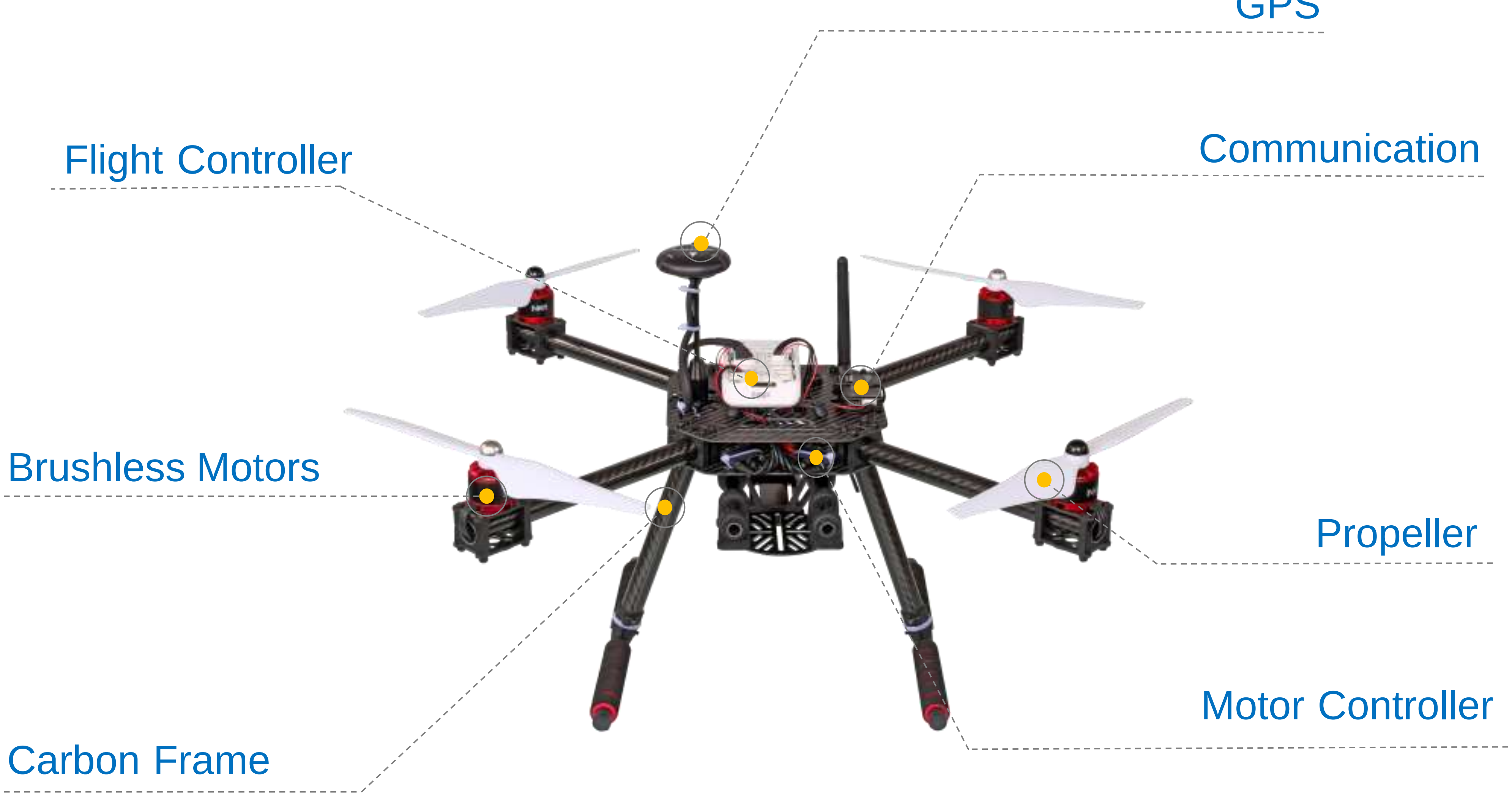
arm

Documentation

www.Hovergames.com

• **Hovergames Drone Gitbook**	https://nxp.gitbook.io/hovergames/
• **PX4 User Guide**	https://docs.px4.io/en/
• **PX4 Developer Guide**	https://dev.px4.io/en/
• **PX4 Github Firmware**	https://github.com/PX4/Firmware
• **NXP Github Starter Code**	https://github.com/nxphovergames

HoverGames Drone



GPS

Flight Controller

Communication

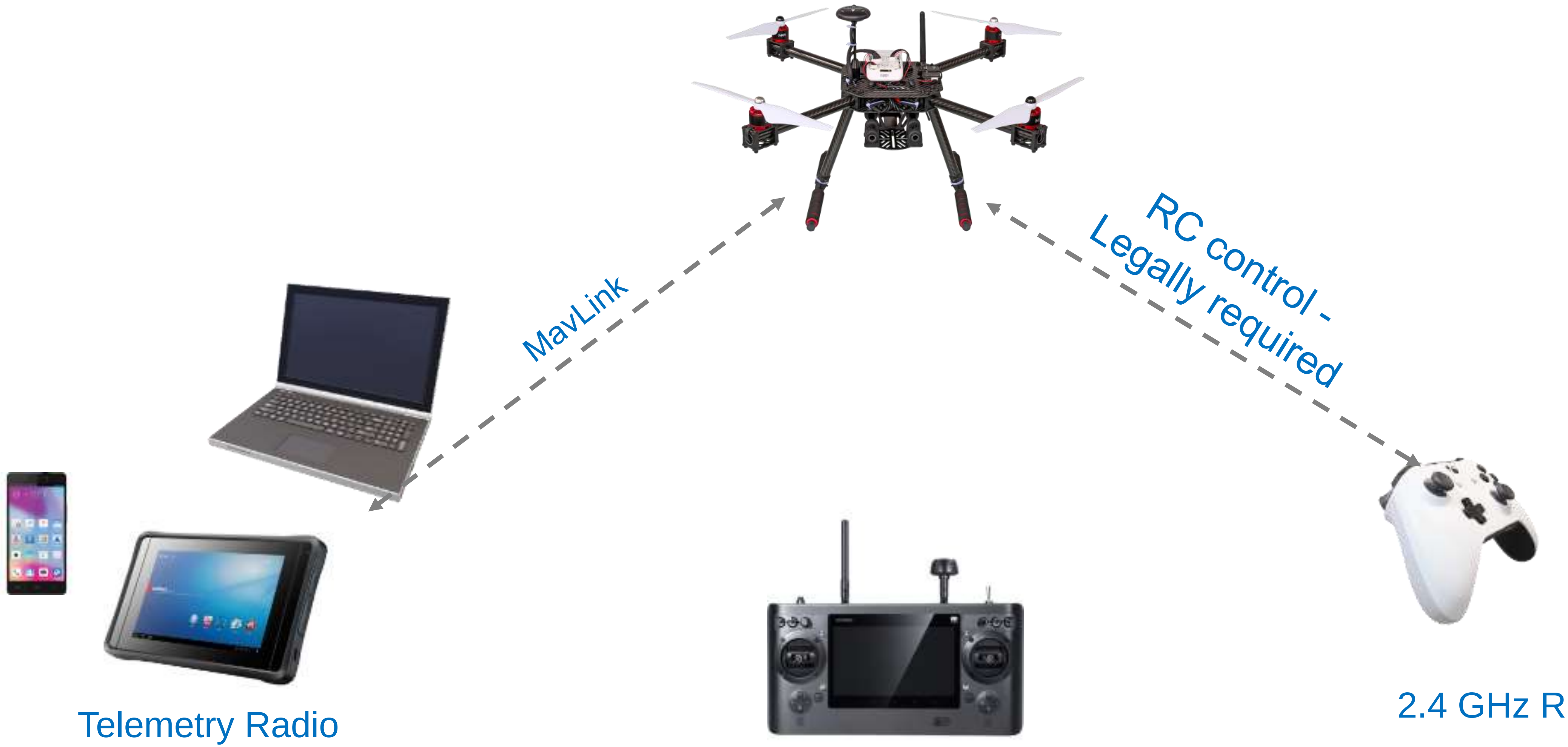
Brushless Motors

Propeller

Motor Controller

Carbon Frame

Communication



MavLink

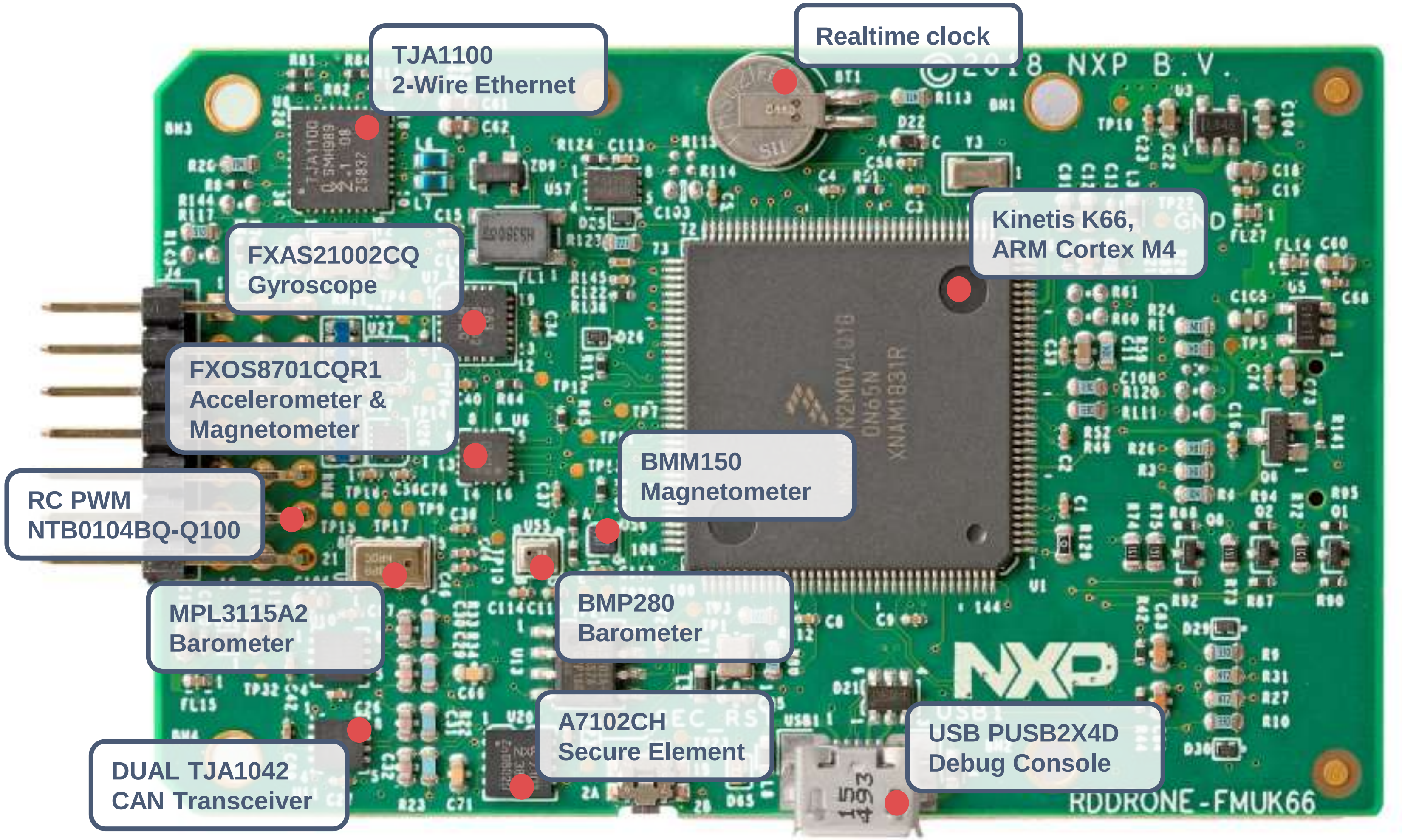
RC control - Legally required

Telemetry Radio
915Mhz/433Mhz

Combination
Controller

2.4 GHz RC

FMU K66



TJA1100
2-Wire Ethernet

Realtime clock

FXAS21002CQ
Gyroscope

Kinetis K66,
ARM Cortex M4

FXOS8701CQR1
Accelerometer &
Magnetometer

BMM150
Magnetometer

RC PWM
NTB0104BQ-Q100

MPL3115A2
Barometer

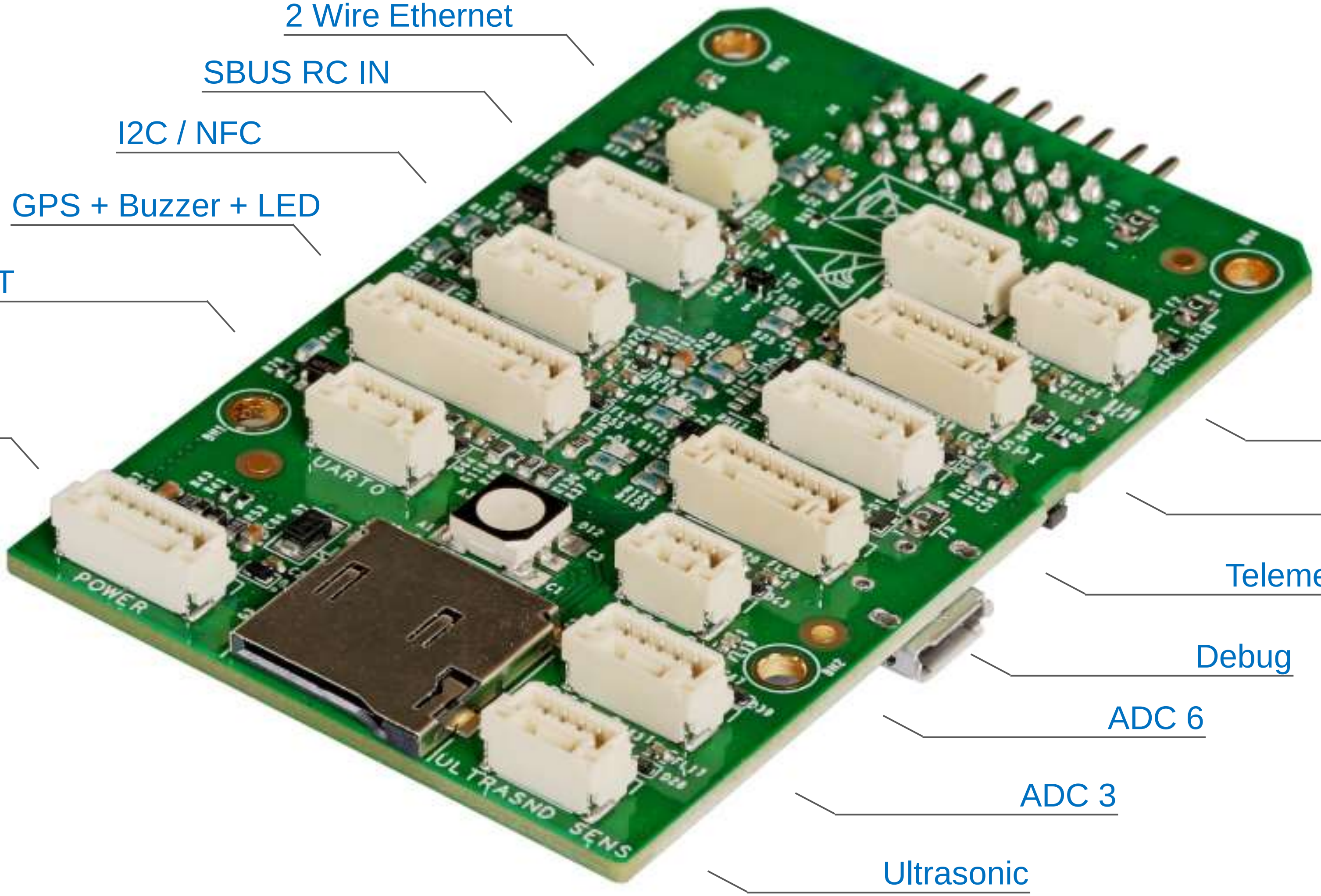
BMP280
Barometer

A7102CH
Secure Element

USB PUSB2X4D
Debug Console

DUAL TJA1042
CAN Transceiver

FMU K66



2 Wire Ethernet

SBUS RC IN

I2C / NFC

GPS + Buzzer + LED

UART

Power

2x Can Bus

SPI

Telemetry

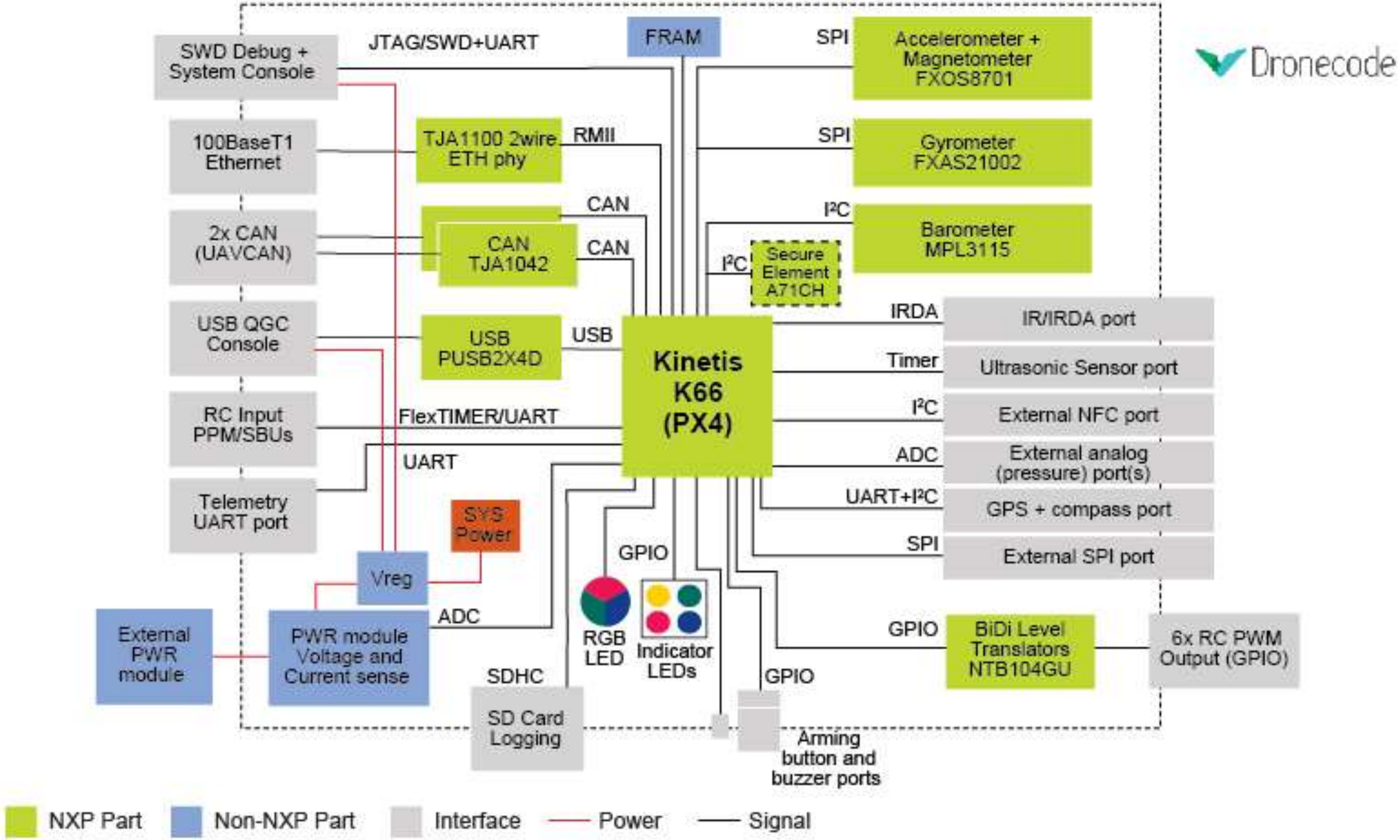
Debug

ADC 6

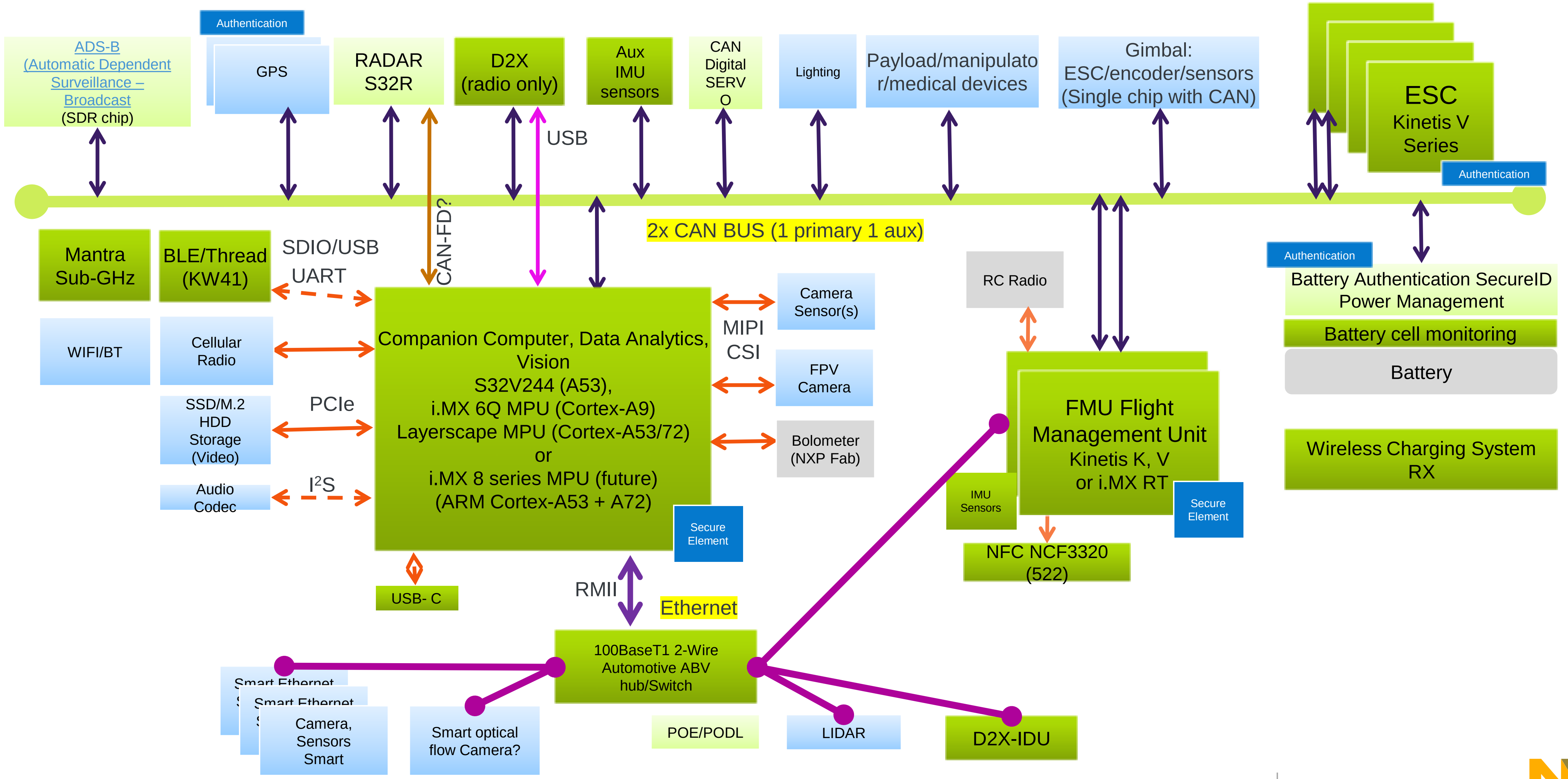
ADC 3

Ultrasonic

RDDRONE-FMUK66 Flight Management Unit



Industrial-Grade UAV – Modular with CAN and Ethernet

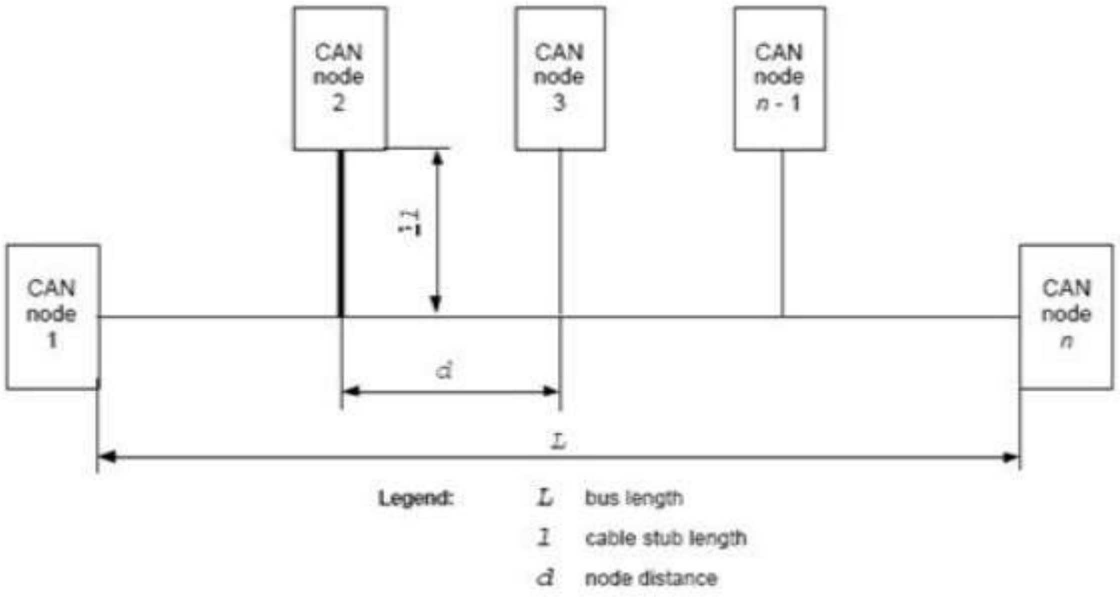


CAN Bus

- It is a BUS, run two for redundancy
- Automotive grade, rugged, robust, high ESD protection, short-proof
- Lightweight connectors and cabling
- Uses address priority, supports many devices
- Hardware enhanced communications

- UAVCAN V1 focus on CAN-FD (5MBps)
- UAVCAN in its infancy – industry wants CAN but lack of hardware at present

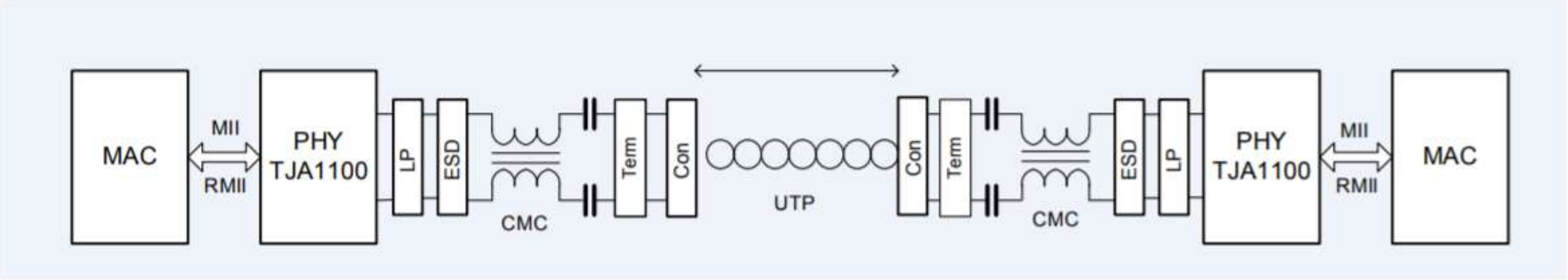
- Look for latest CAN PHYs from NXP
  - Stinger, with hardware backed security. White/Blacklist
  - Enhanced bus matching



ESC and Servos not limited by number of PWM ports.

Data can be shared at the bus level (i.e multiple GPS modules)

100BaseT1 “2 wire ethernet”



- Automotive Rugged, robust, high ESD
- Lightweight connectors, wires, no magnetics
- Faster Gbit , 2.5 Gbit coming
- 15 meter distance
- Automotive ethernet Switch available
- Still regular ethernet - media conversion by switch or back to back PHYs
- Attractive for high speed IP/Socket programming

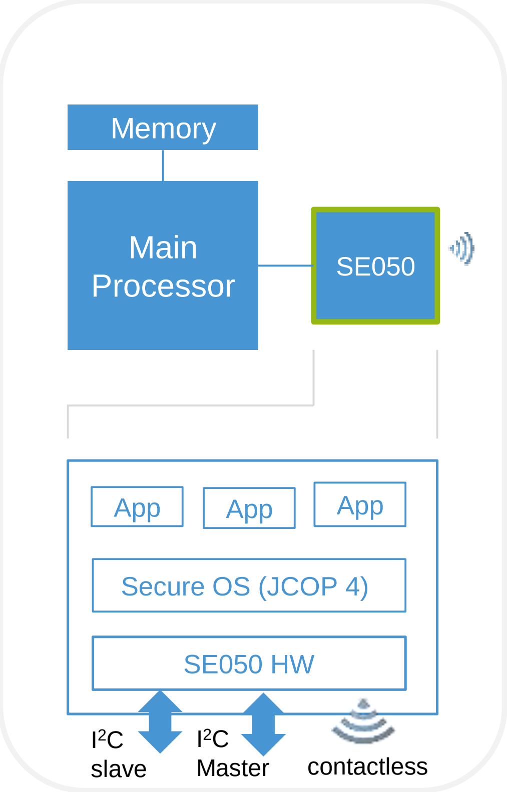
- Higher bandwidth cameras or sensors
- Standard IP connection between FMU and Companion computer
- Tethered operation

Secure Element A71CH / SE050

- Embedded Secure Element
  - Discrete HW Tamper resistant security component
  - State-of-the-art security, certified
  - Dedicated environment to host security functions (isolation)
  - Companion chip to any type of MCU, MPU and AP

- Secure sub-system on IoT Edge:
  - Drone Authentication, Subsystem Authentication
  - Secure hardware encryption
  - Physical security
  - Secure key store
  - UTM Airspace

Also A1006/1007 for authentication only i.e battery packs



- FMU
- BMS
- ESC
- Payloads
- Drone to drone
- Drone to ground station
- Drone to OTHER e-identification (BLE long range?)

PX4 is an Integrated Software Ecosystem

PX4 autopilot

Vehicle

Firmware

Communication

MAVLINK

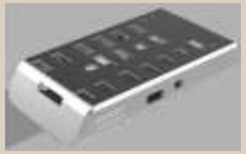
MICRO AIR VEHICLE COMMUNICATION PROTOCOL

Ground Control Station

QGroundControl

Hardware

pixhawk



NXP - NXPhlite

Intel® Aero

Autonomy

Vision based localization and avoidance

Middleware ROS2

2

API

<dronecode/sdk>

Leading Products and Dev Platforms Based on PX4

YUNEEC

YUNEEC

IMPOSSIBLE AEROSPACE

UVify

FREEFLY

NXP

DHL

amazon Prime Air

QUALCOMM

INSPIRED FLIGHT

TEAL

SONY

sentera

VERTICAL technologies

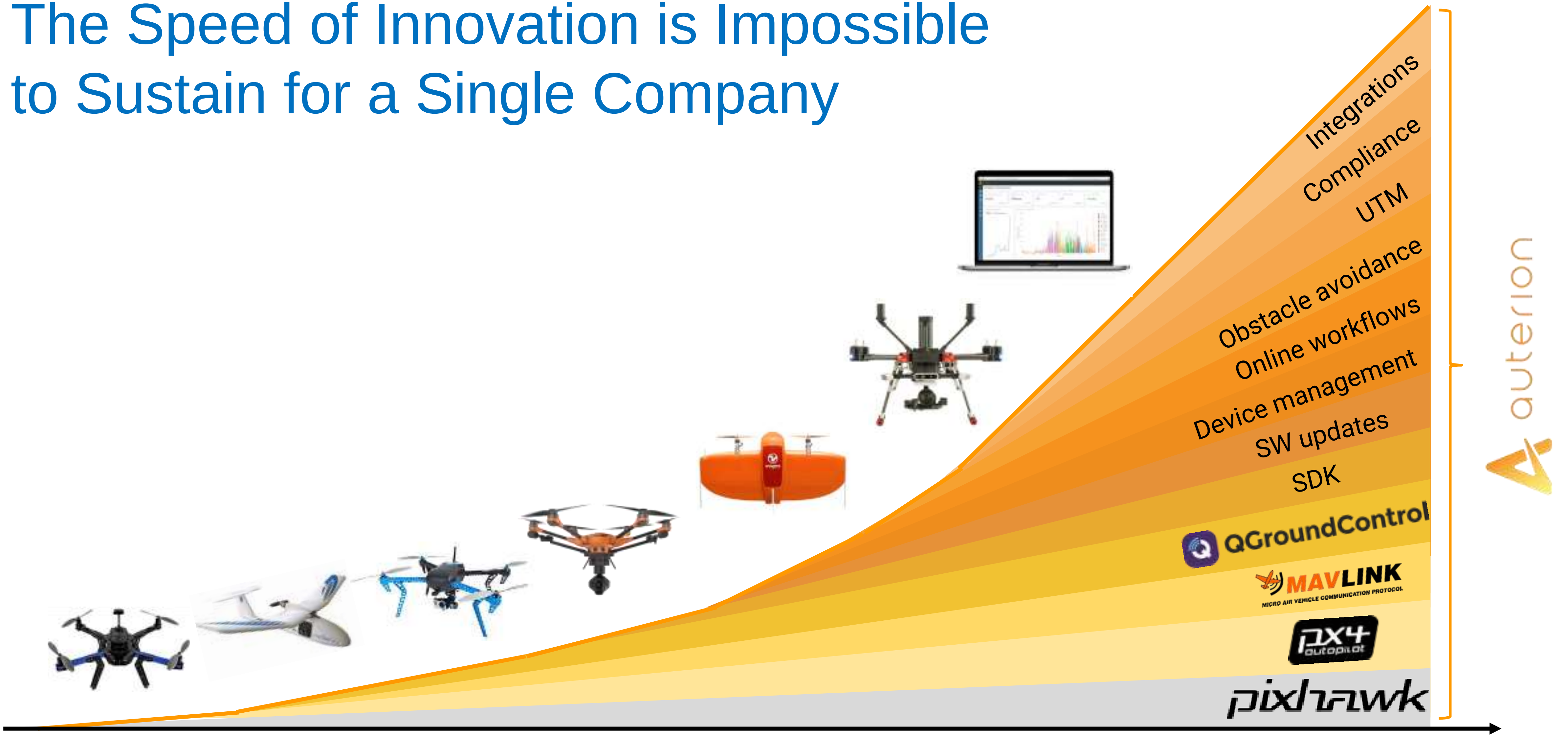
ALTi

FlightWave

AIRDOG

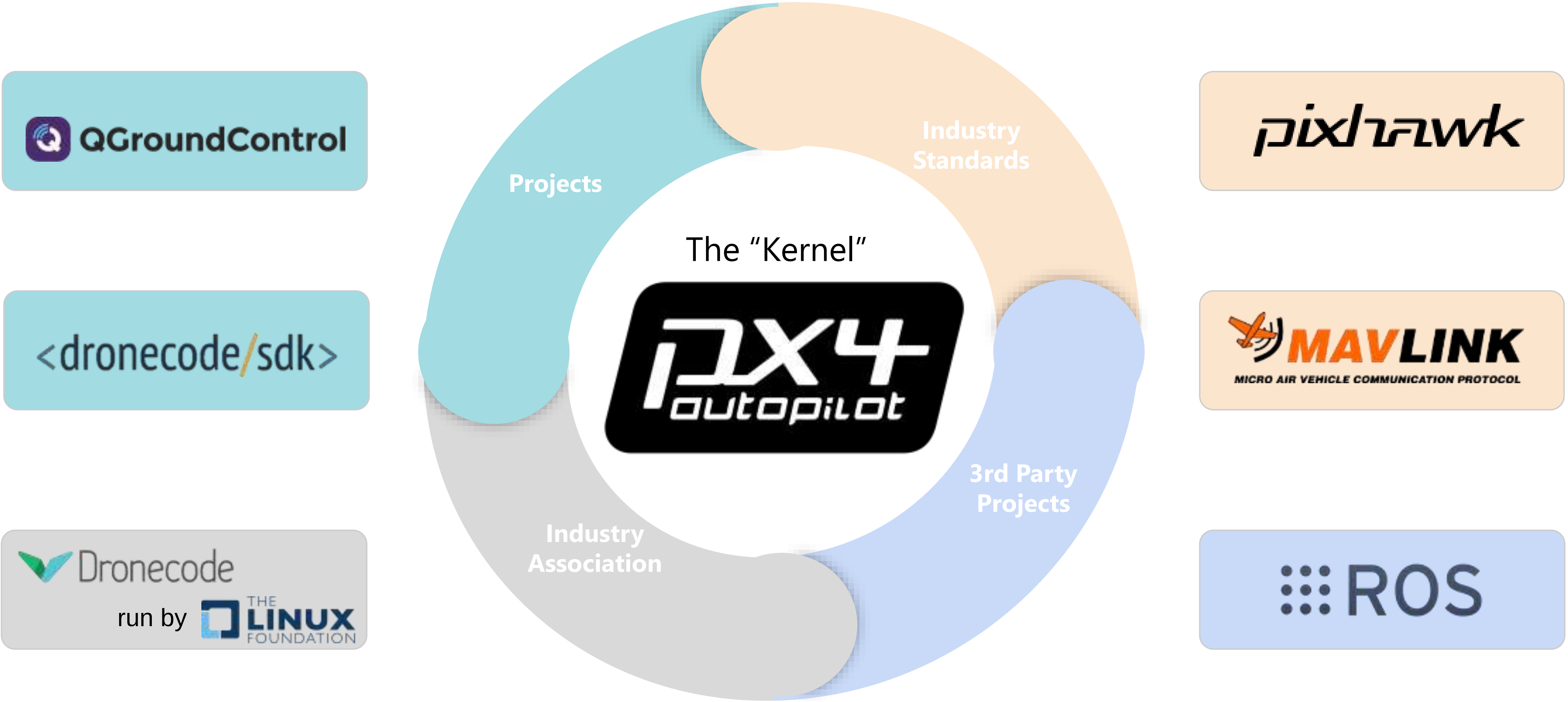
wingtra

The Speed of Innovation is Impossible to Sustain for a Single Company



2008

What is the Relationship Between the Parts?



QGroundControl

<dronecode/sdk>

Dronecode run by THE LINUX FOUNDATION

Projects

Industry Standards

The "Kernel"

PX4 autopilot

3rd Party Projects

Industry Association

pixhawk

MAVLINK MICRO AIR VEHICLE COMMUNICATION PROTOCOL

ROS

The Business of Open Source

Auterion Enterprise PX4: the Operating System for Drones

Auterion Enterprise PX4 software that runs on the drone.

It enables:

- Drones to fly safely and autonomously
- Detect and avoid obstacles, and prevent collisions
- Positioning with RTK GPS accuracy
- Online workflows (data upload) with connection to LTE

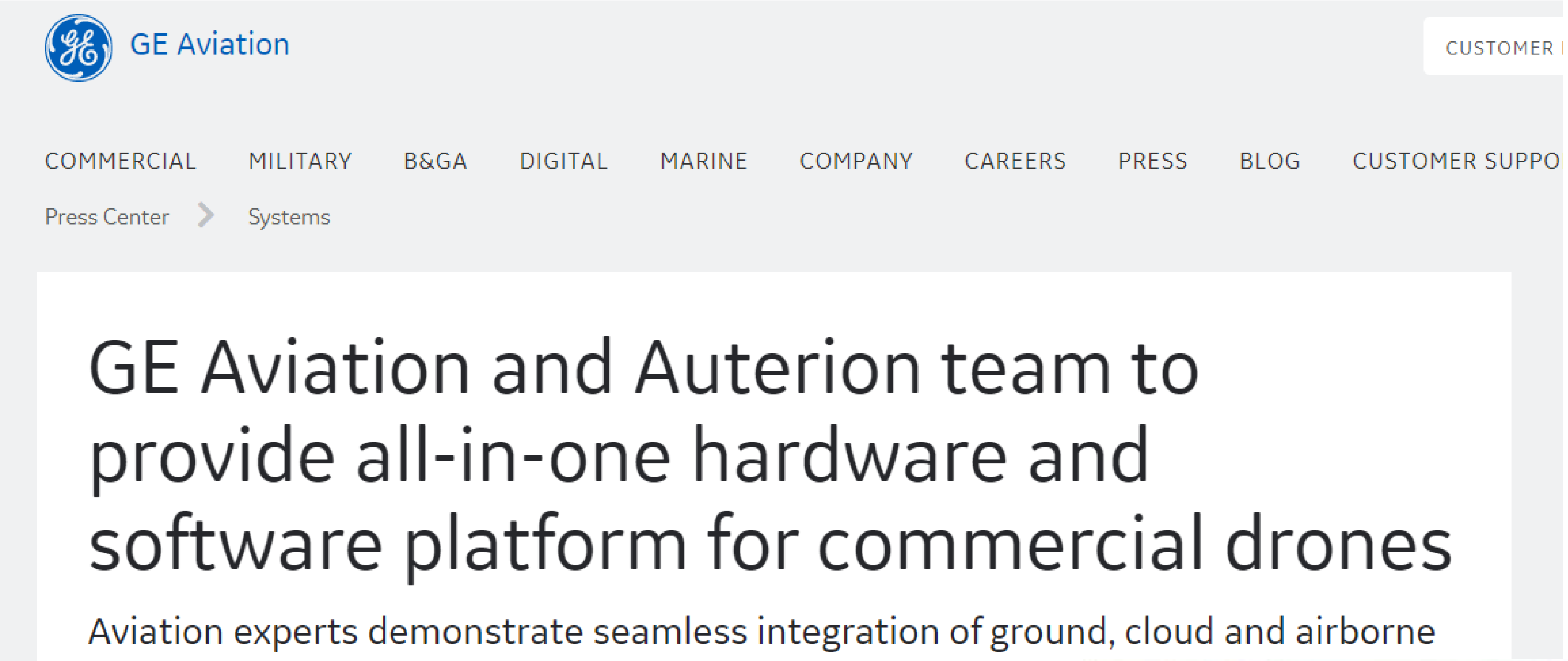
1 Auterion Enterprise PX4

2 Auterion Ground Station

3 Auterion Cloud Insights

Auterion vs. Open Source

		Auterion enterprise distribution	Community/ Open source PX4
Flight controller	PX4 autopilot	✓	✓
Ground station	Q	✓	✓
Compatible hardware	pixhawk	✓	✓
Obstacle avoidance		✓	✓
Device / Fleet management		✓	
Compliance and regulatory approval		✓	
Enterprise Support and SW updates		✓	
Cybersecurity hardening		✓	



GE Aviation

CUSTOMER I

COMMERCIAL MILITARY B&GA DIGITAL MARINE COMPANY CAREERS PRESS BLOG CUSTOMER SUPPO

Press Center > Systems

GE Aviation and Auterion team to provide all-in-one hardware and software platform for commercial drones

Aviation experts demonstrate seamless integration of ground, cloud and airborne components

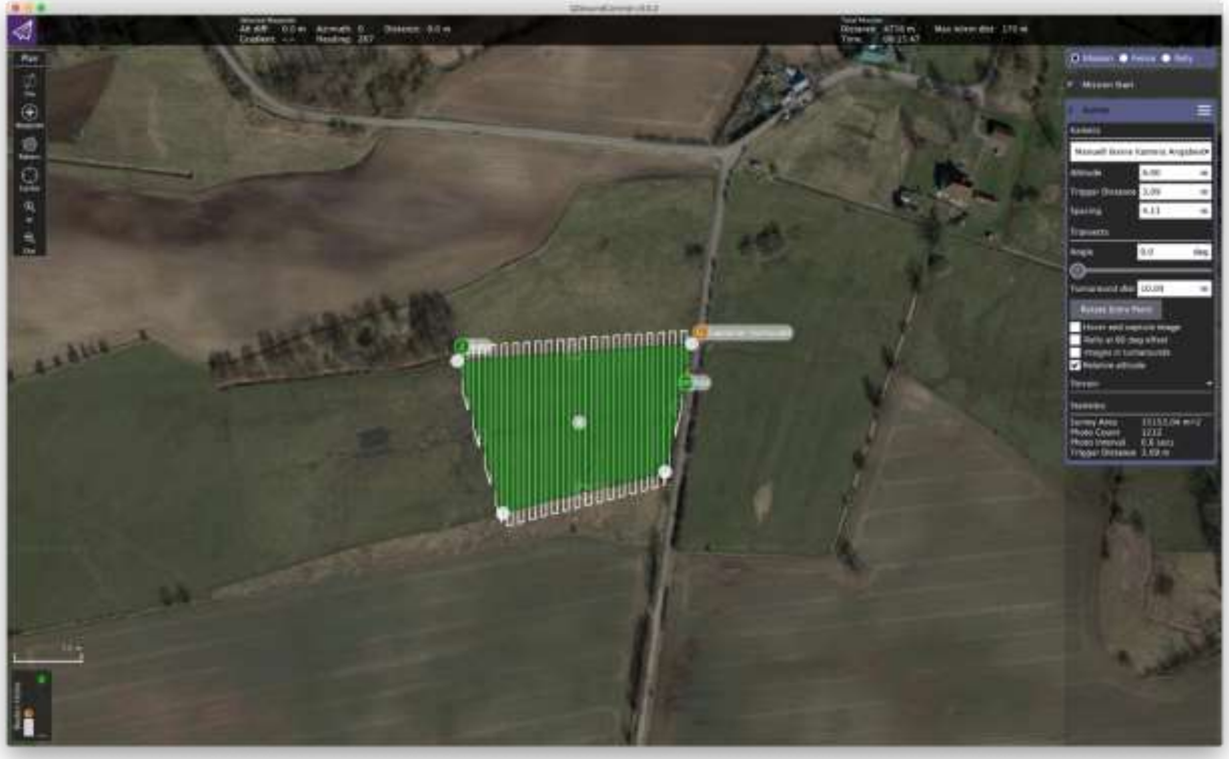
April 29, 2019

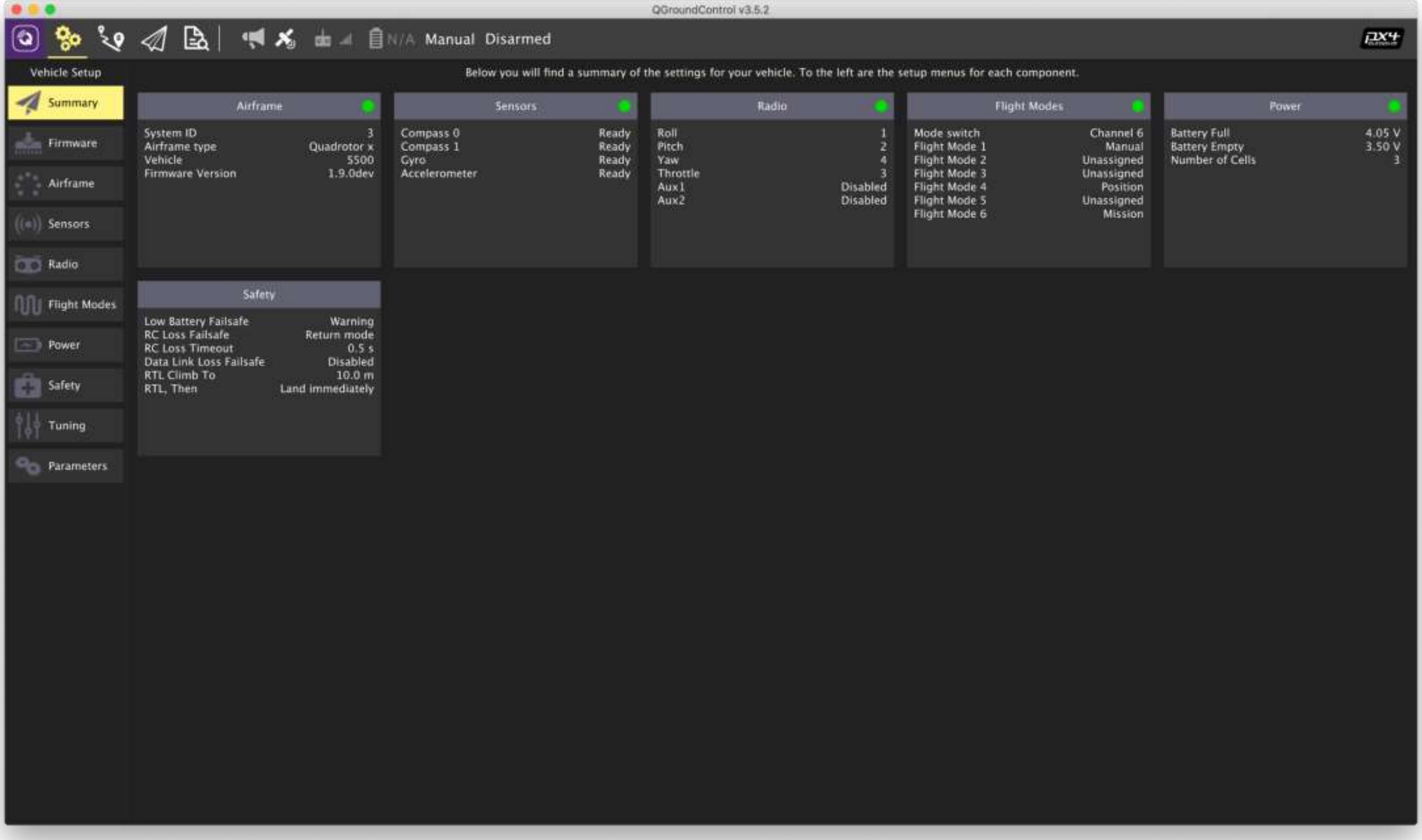
Today, GE Aviation and Auterion announce the integration of the Auterion Enterprise Aviation's Unmanned Aircraft System avionics platform. They show their commitmer agreement to provide a comprehensive hardware and software solution for drone mai seeking to enable commercial drone operations at scale.

QGroundcontrol

QGroundcontrol

- State of the art software runs on Android, iOS, Windows, Linux and Mac OS.
- User friendly for flying and mission planning
- Survey, mapping support
- Digital video streaming support
- Enables firmware customization
- Simple airframe setup, including firmware, joysticks, sensors and flight modes.





Below you will find a summary of the settings for your vehicle. To the left are the setup menus for each component.

Airframe

System ID	3
Airframe type	Quadrotor x
Vehicle	5500
Firmware Version	1.9.0dev

Sensors

Compass 0	Ready
Compass 1	Ready
Gyro	Ready
Accelerometer	Ready

Radio

Roll	1
Pitch	2
Yaw	4
Throttle	3
Aux1	Disabled
Aux2	Disabled

Flight Modes

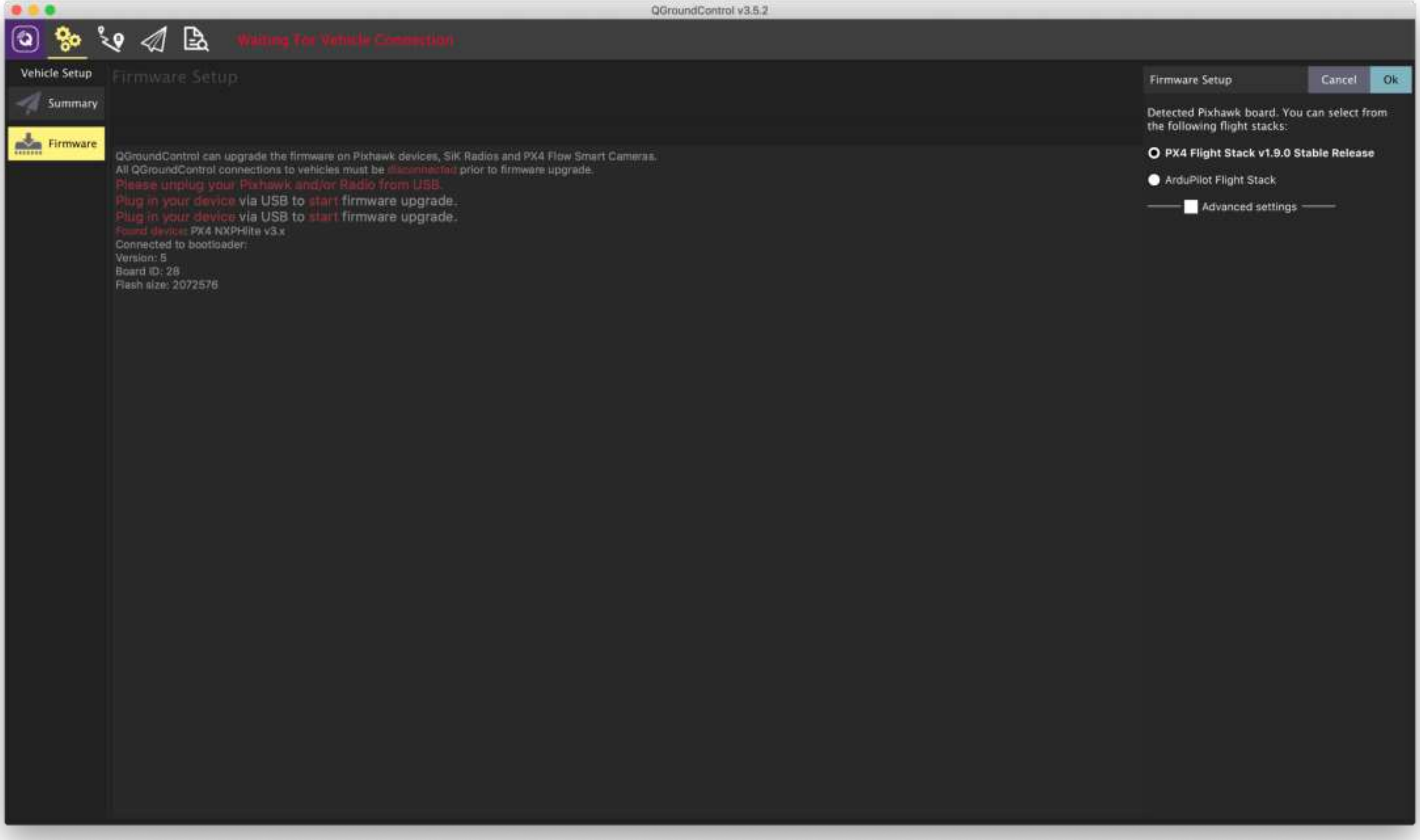
Mode switch	Channel 6
Flight Mode 1	Manual
Flight Mode 2	Unassigned
Flight Mode 3	Unassigned
Flight Mode 4	Position
Flight Mode 5	Unassigned
Flight Mode 6	Mission

Power

Battery Full	4.05 V
Battery Empty	3.50 V
Number of Cells	3

Safety

Low Battery Failsafe	Warning
RC Loss Failsafe	Return mode
RC Loss Timeout	0.5 s
Data Link Loss Failsafe	Disabled
RTL Climb To	10.0 m
RTL, Then	Land immediately



```
QGroundControl can upgrade the firmware on Pixhawk devices, SiK Radios and PX4 Flow Smart Cameras.
All QGroundControl connections to vehicles must be disconnected prior to firmware upgrade.
Please unplug your Pixhawk and/or Radio from USB.
Plug in your device via USB to start firmware upgrade.
Plug in your device via USB to start firmware upgrade.
Found device: PX4 NXPHlite v3.x
Connected to bootloader:
Version: 5
Board ID: 28
Flash size: 2072576
```

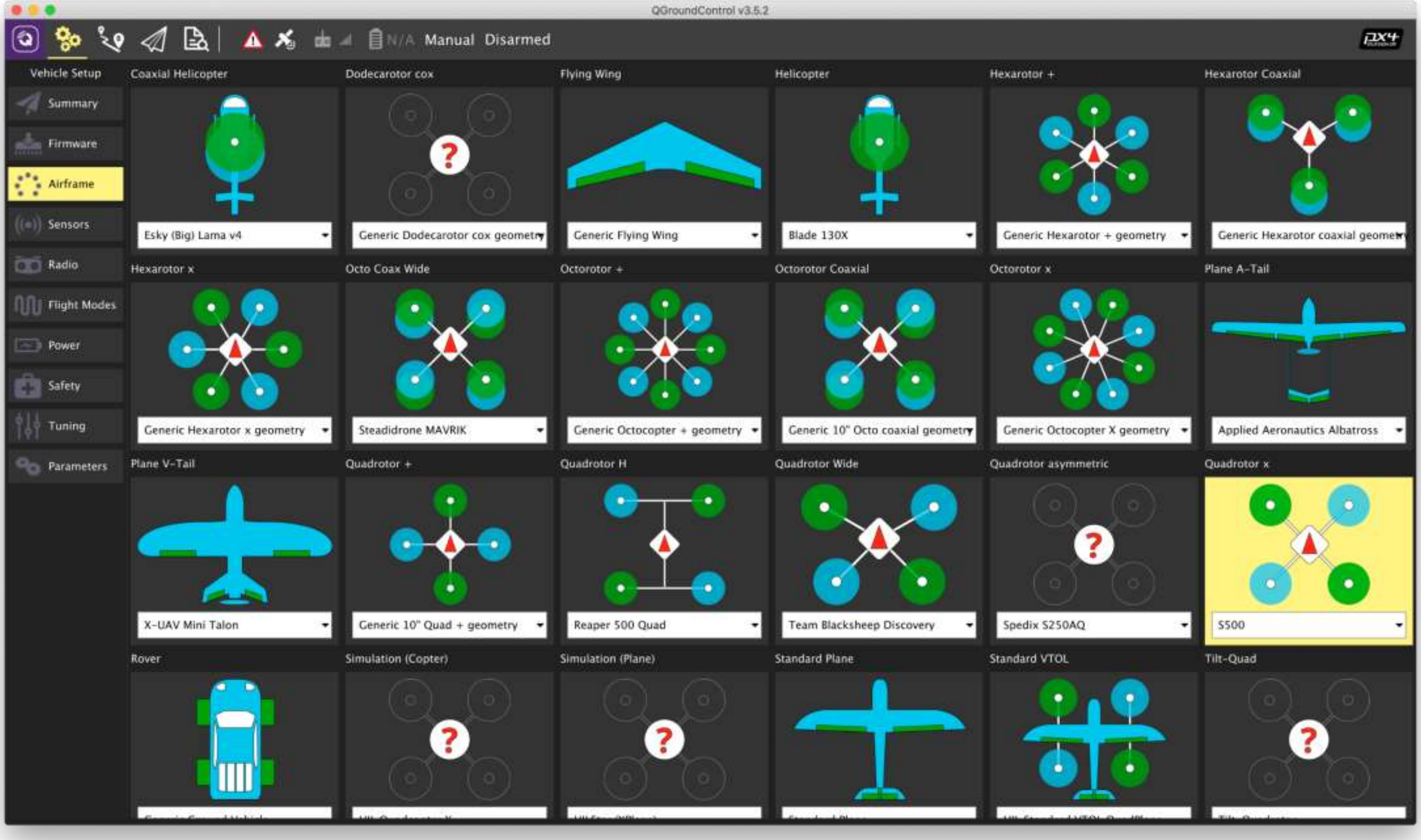
Firmware Setup

Detected Pixhawk board. You can select from the following flight stacks:

● PX4 Flight Stack v1.9.0 Stable Release

○ ArduPilot Flight Stack

□ Advanced settings



Coaxial Helicopter
Esky (Big) Lama v4

Dodecarotor cox
Generic Dodecarotor cox geometry

Flying Wing
Generic Flying Wing

Helicopter
Blade 130X

Hexarotor +
Generic Hexarotor + geometry

Hexarotor Coaxial
Generic Hexarotor coaxial geometry

Hexarotor x
Generic Hexarotor x geometry

Octo Coax Wide
Steadidrone MAVRIK

Octorotor +
Generic Octocopter + geometry

Octorotor Coaxial
Generic 10" Octo coaxial geometry

Octorotor x
Generic Octocopter X geometry

Plane A-Tail
Applied Aeronautics Albatross

Plane V-Tail
X-UAV Mini Talon

Quadrotor +
Generic 10" Quad + geometry

Quadrotor H
Reaper 500 Quad

Quadrotor Wide
Team Blacksheep Discovery

Quadrotor asymmetric
Spedix S250AQ

Quadrotor x
S500

Rover

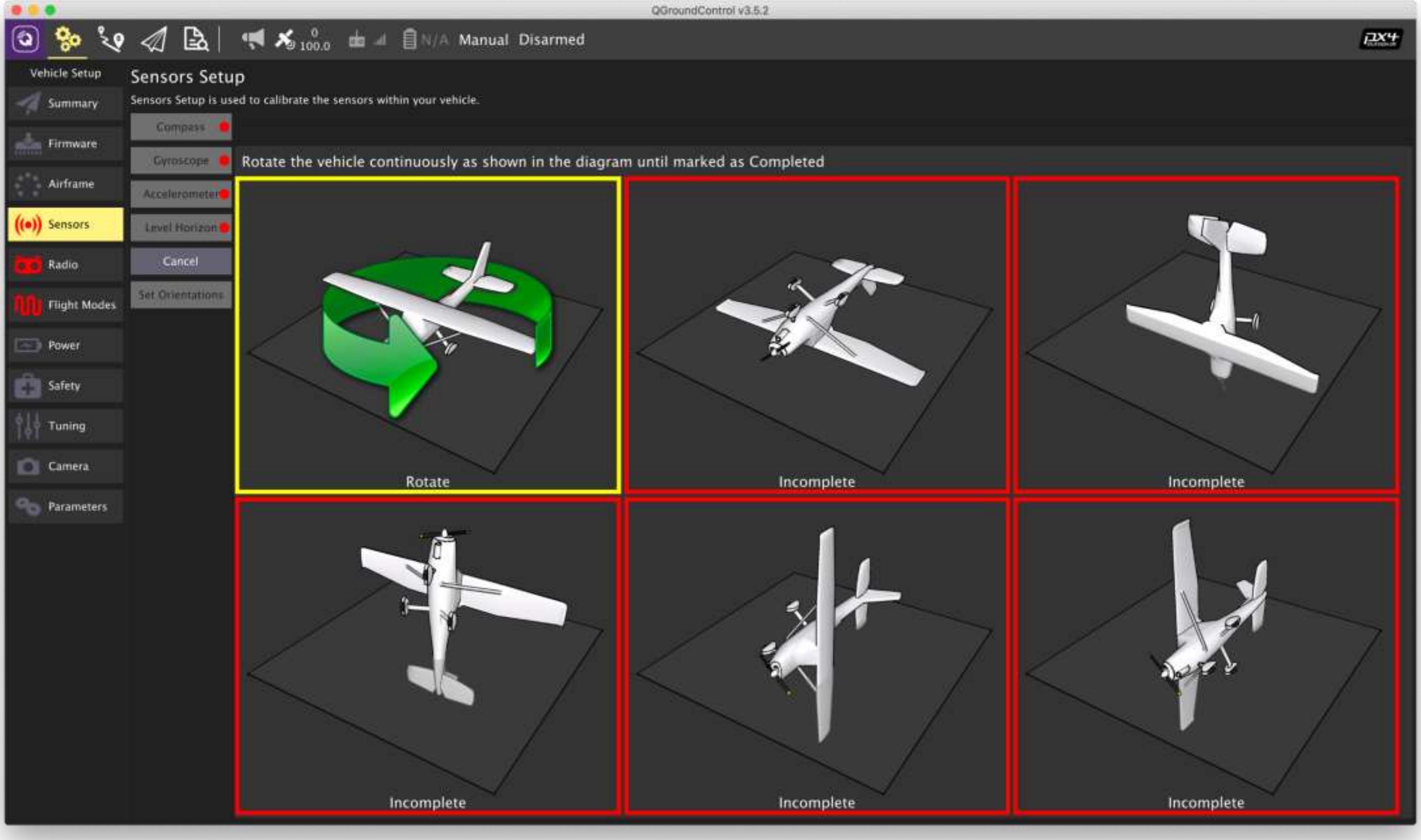
Simulation (Copter)

Simulation (Plane)

Standard Plane

Standard VTOL

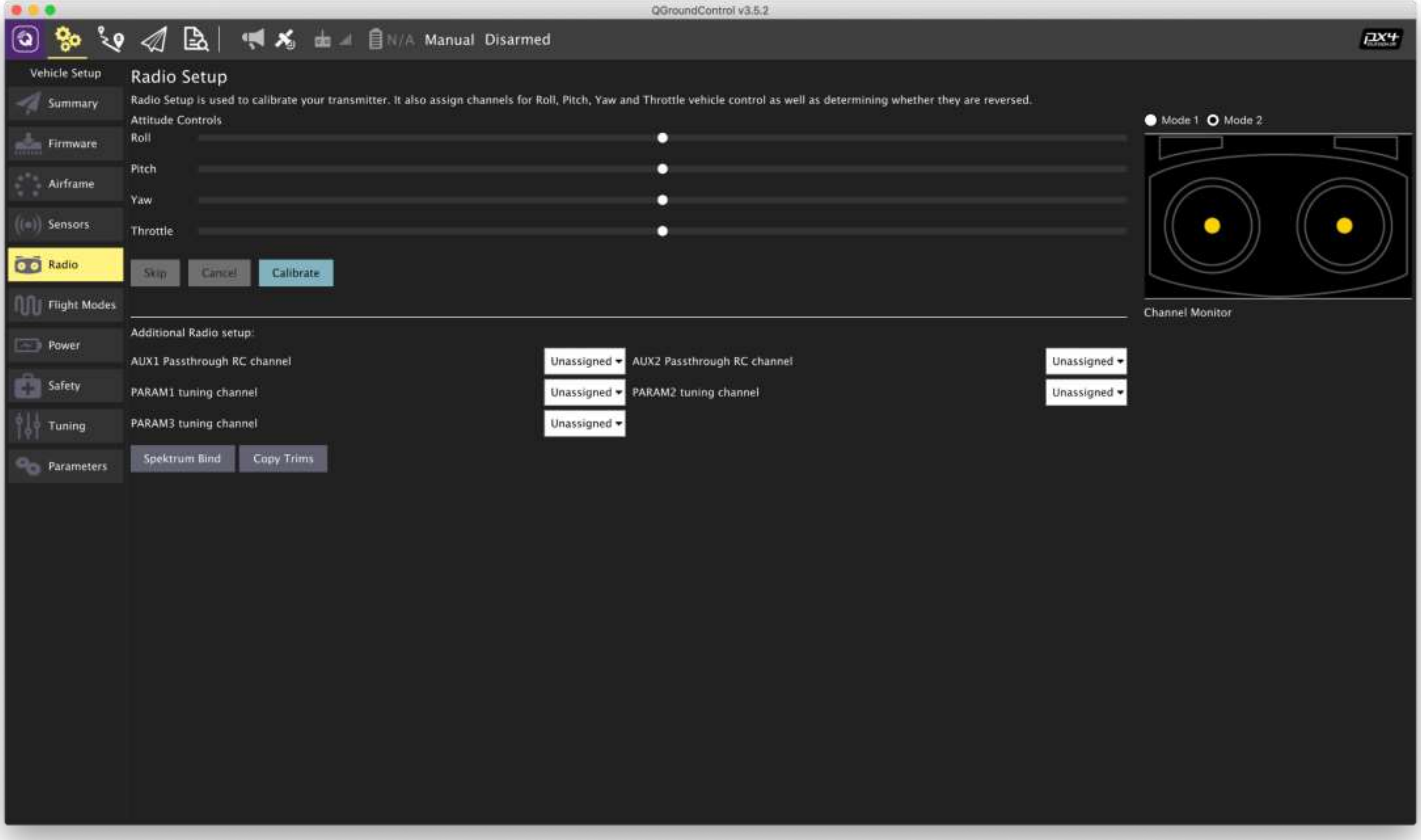
Tilt-Quad



Sensors Setup

Sensors Setup is used to calibrate the sensors within your vehicle.

Rotate the vehicle continuously as shown in the diagram until marked as Completed



Radio Setup

Radio Setup is used to calibrate your transmitter. It also assign channels for Roll, Pitch, Yaw and Throttle vehicle control as well as determining whether they are reversed.

Attitude Controls

Roll

Pitch

Yaw

Throttle

● Mode 1 ○ Mode 2

Channel Monitor

Additional Radio setup:

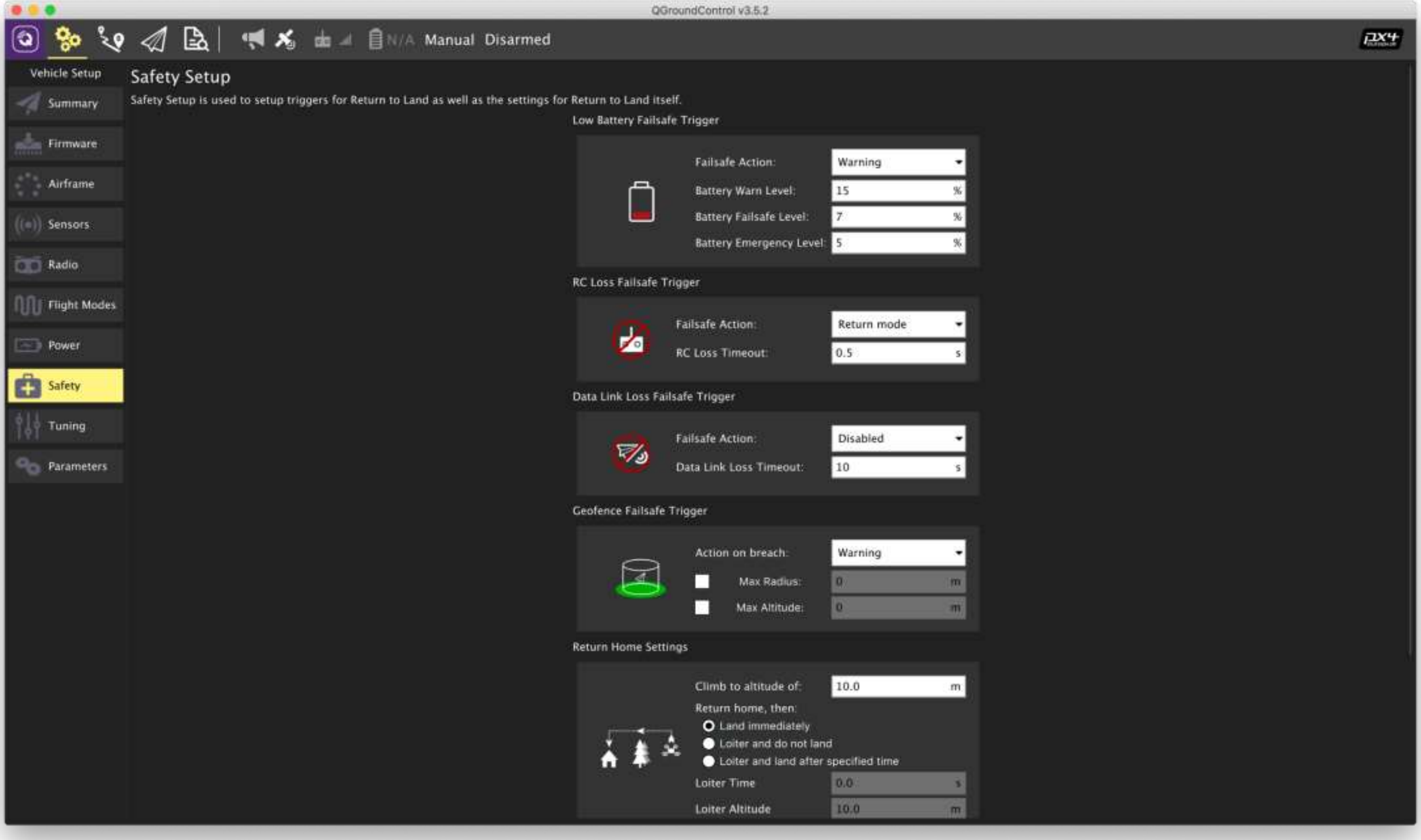
AUX1 Passthrough RC channel: Unassigned

AUX2 Passthrough RC channel: Unassigned

PARAM1 tuning channel: Unassigned

PARAM2 tuning channel: Unassigned

PARAM3 tuning channel: Unassigned



Safety Setup

Safety Setup is used to setup triggers for Return to Land as well as the settings for Return to Land itself.

Low Battery Failsafe Trigger

Failsafe Action:	Warning
Battery Warn Level:	15 %
Battery Failsafe Level:	7 %
Battery Emergency Level:	5 %

RC Loss Failsafe Trigger

Failsafe Action:	Return mode
RC Loss Timeout:	0.5 s

Data Link Loss Failsafe Trigger

Failsafe Action:	Disabled
Data Link Loss Timeout:	10 s

Geofence Failsafe Trigger

Action on breach:	Warning
□ Max Radius:	0 m
□ Max Altitude:	0 m

Return Home Settings

Climb to altitude of: 10.0 m

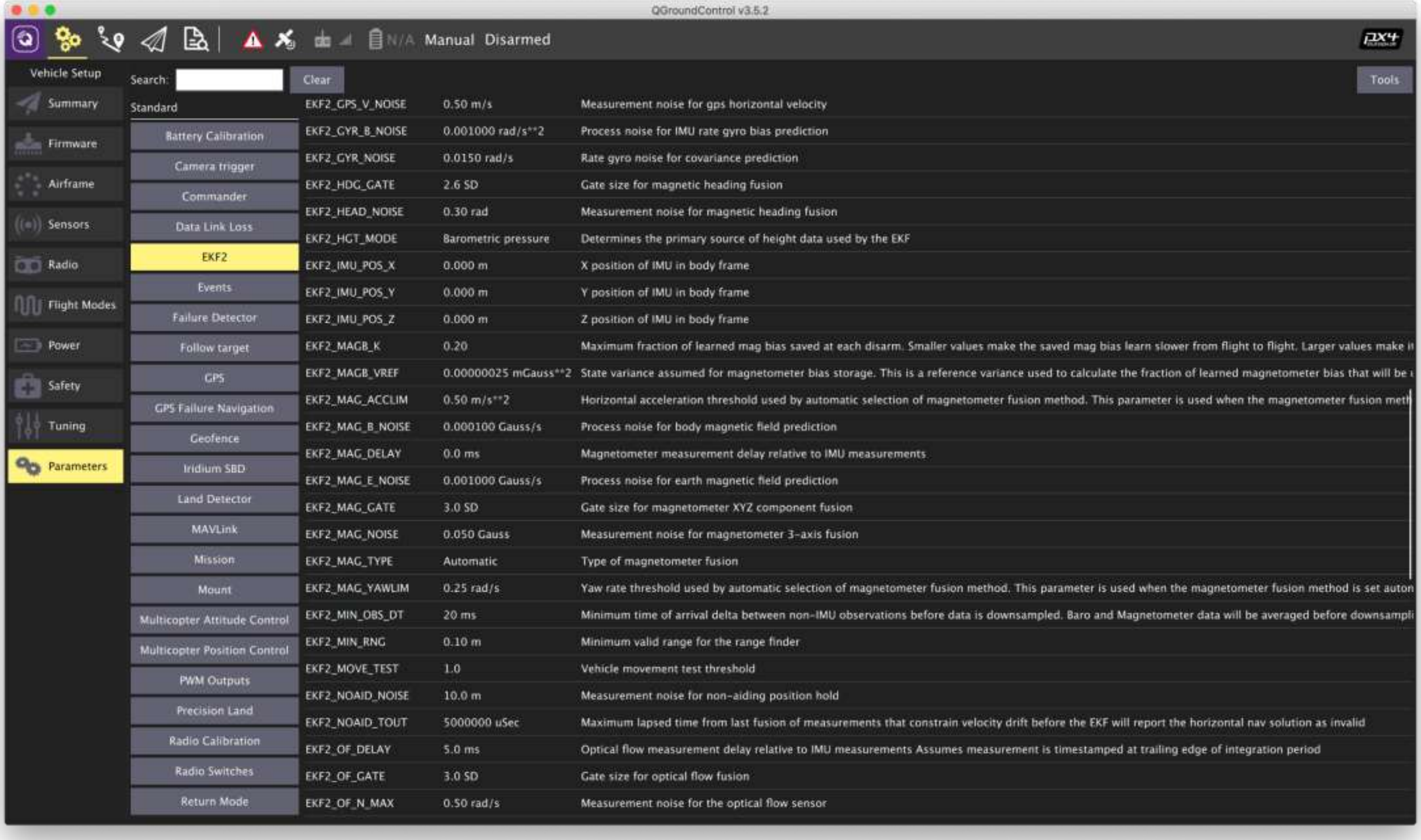
Return home, then:

○ Land immediately

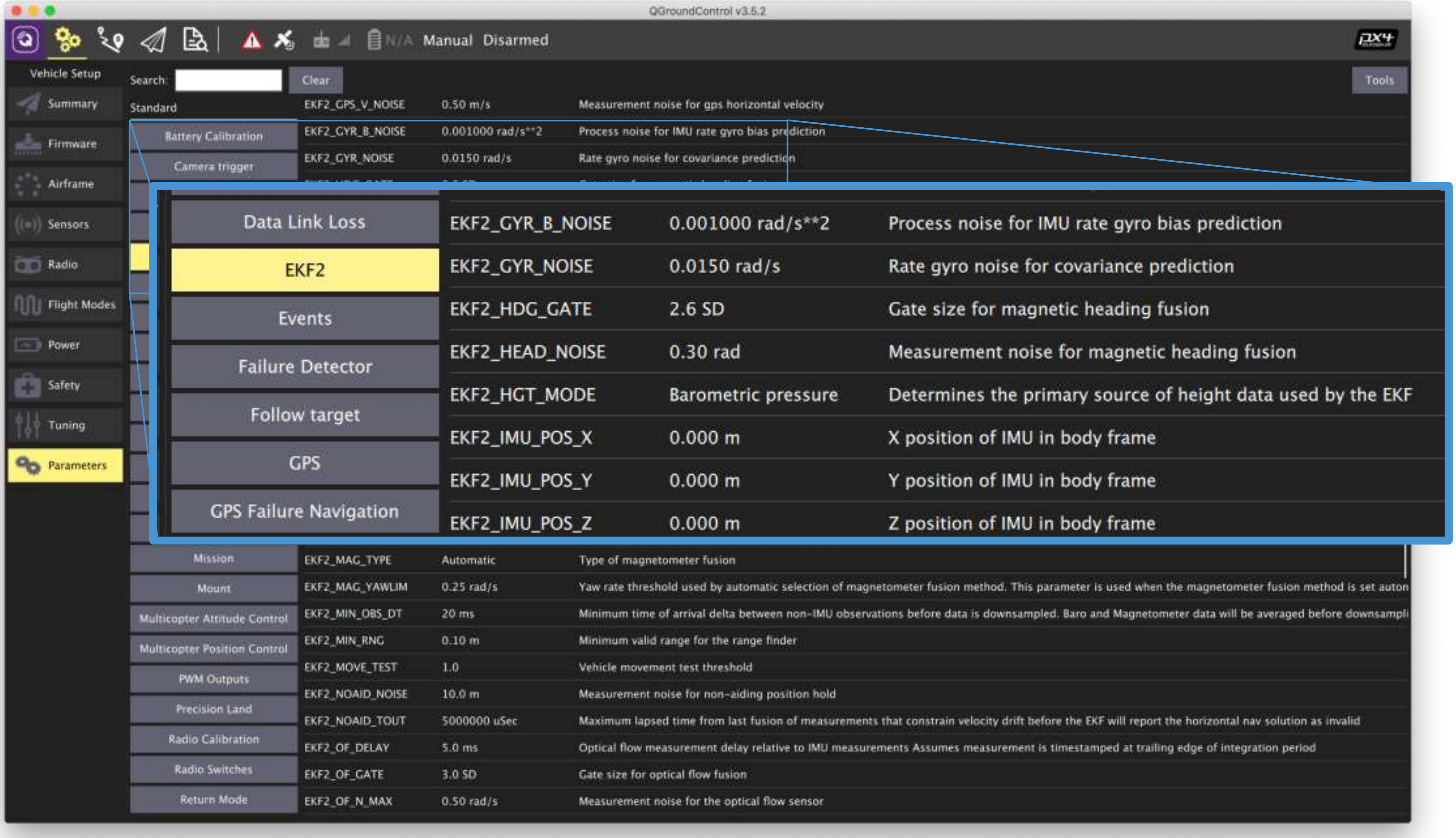
● Loiter and do not land

● Loiter and land after specified time

Loiter Time	0.0 s
Loiter Altitude	10.0 m



EKF2_GPS_V_NOISE	0.50 m/s	Measurement noise for gps horizontal velocity
EKF2_GYR_B_NOISE	0.001000 rad/s**2	Process noise for IMU rate gyro bias prediction
EKF2_GYR_NOISE	0.0150 rad/s	Rate gyro noise for covariance prediction
EKF2_HDG_GATE	2.6 SD	Gate size for magnetic heading fusion
EKF2_HEAD_NOISE	0.30 rad	Measurement noise for magnetic heading fusion
EKF2_HGT_MODE	Barometric pressure	Determines the primary source of height data used by the EKF
EKF2_IMU_POS_X	0.000 m	X position of IMU in body frame
EKF2_IMU_POS_Y	0.000 m	Y position of IMU in body frame
EKF2_IMU_POS_Z	0.000 m	Z position of IMU in body frame
EKF2_MAGB_K	0.20	Maximum fraction of learned mag bias saved at each disarm. Smaller values make the saved mag bias learn slower from flight to flight. Larger values make i
EKF2_MAGB_VREF	0.00000025 mGauss**2	State variance assumed for magnetometer bias storage. This is a reference variance used to calculate the fraction of learned magnetometer bias that will be
EKF2_MAG_ACCLIM	0.50 m/s**2	Horizontal acceleration threshold used by automatic selection of magnetometer fusion method. This parameter is used when the magnetometer fusion meth
EKF2_MAG_B_NOISE	0.000100 Gauss/s	Process noise for body magnetic field prediction
EKF2_MAG_DELAY	0.0 ms	Magnetometer measurement delay relative to IMU measurements
EKF2_MAG_E_NOISE	0.001000 Gauss/s	Process noise for earth magnetic field prediction
EKF2_MAG_GATE	3.0 SD	Gate size for magnetometer XYZ component fusion
EKF2_MAG_NOISE	0.050 Gauss	Measurement noise for magnetometer 3-axis fusion
EKF2_MAG_TYPE	Automatic	Type of magnetometer fusion
EKF2_MAG_YAWLIM	0.25 rad/s	Yaw rate threshold used by automatic selection of magnetometer fusion method. This parameter is used when the magnetometer fusion method is set auton
EKF2_MIN_OBS_DT	20 ms	Minimum time of arrival delta between non-IMU observations before data is downsampled. Baro and Magnetometer data will be averaged before downsampli
EKF2_MIN_RNG	0.10 m	Minimum valid range for the range finder
EKF2_MOVE_TEST	1.0	Vehicle movement test threshold
EKF2_NOAID_NOISE	10.0 m	Measurement noise for non-aiding position hold
EKF2_NOAID_TOUT	5000000 uSec	Maximum lapsed time from last fusion of measurements that constrain velocity drift before the EKF will report the horizontal nav solution as invalid
EKF2_OF_DELAY	5.0 ms	Optical flow measurement delay relative to IMU measurements Assumes measurement is timestamped at trailing edge of integration period
EKF2_OF_GATE	3.0 SD	Gate size for optical flow fusion
EKF2_OF_N_MAX	0.50 rad/s	Measurement noise for the optical flow sensor



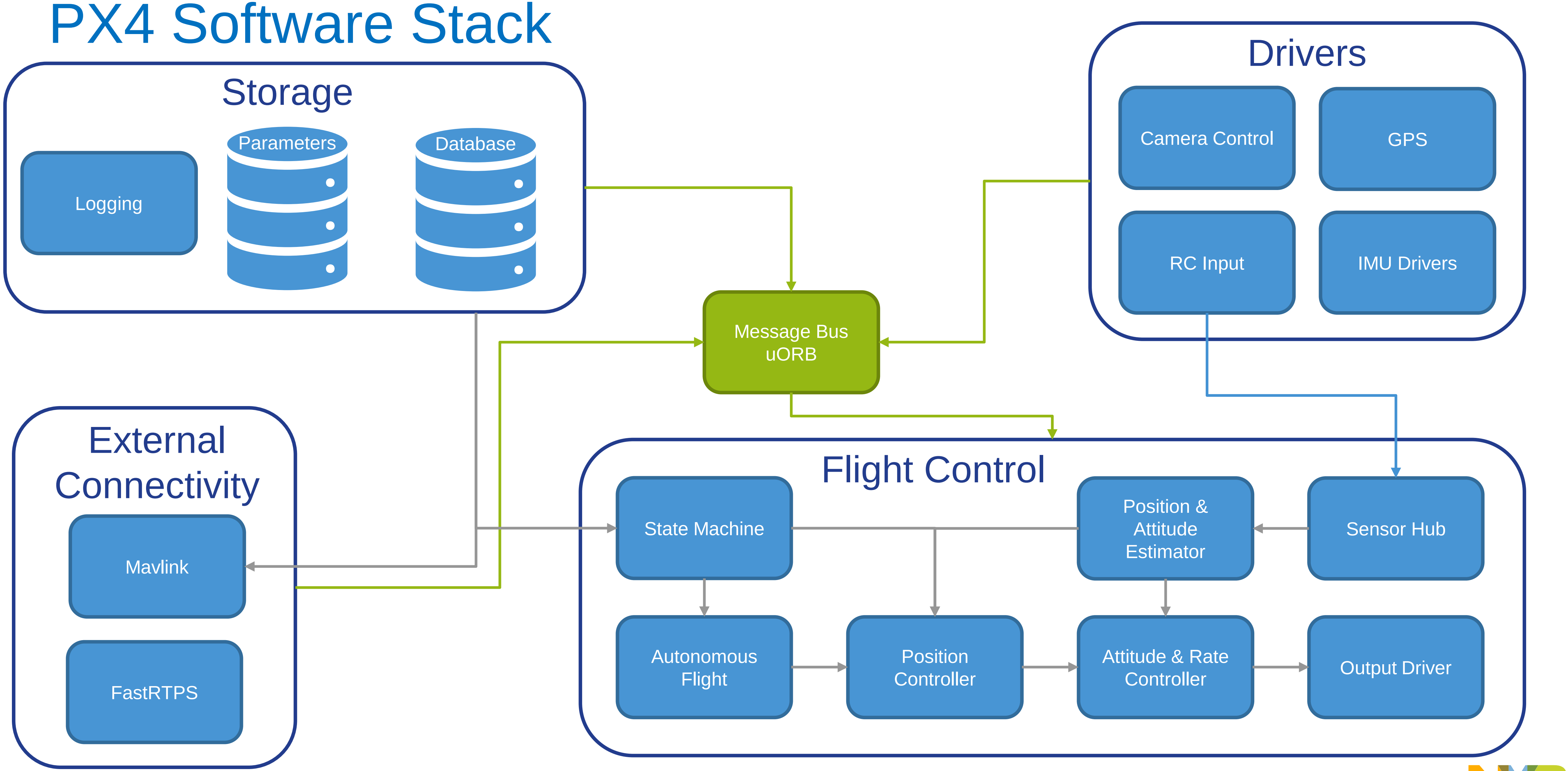
EKF2_GPS_V_NOISE	0.50 m/s	Measurement noise for gps horizontal velocity
EKF2_GYR_B_NOISE	0.001000 rad/s**2	Process noise for IMU rate gyro bias prediction
EKF2_GYR_NOISE	0.0150 rad/s	Rate gyro noise for covariance prediction

EKF2_GYR_B_NOISE	0.001000 rad/s**2	Process noise for IMU rate gyro bias prediction
EKF2_GYR_NOISE	0.0150 rad/s	Rate gyro noise for covariance prediction
EKF2_HDG_GATE	2.6 SD	Gate size for magnetic heading fusion
EKF2_HEAD_NOISE	0.30 rad	Measurement noise for magnetic heading fusion
EKF2_HGT_MODE	Barometric pressure	Determines the primary source of height data used by the EKF
EKF2_IMU_POS_X	0.000 m	X position of IMU in body frame
EKF2_IMU_POS_Y	0.000 m	Y position of IMU in body frame
EKF2_IMU_POS_Z	0.000 m	Z position of IMU in body frame

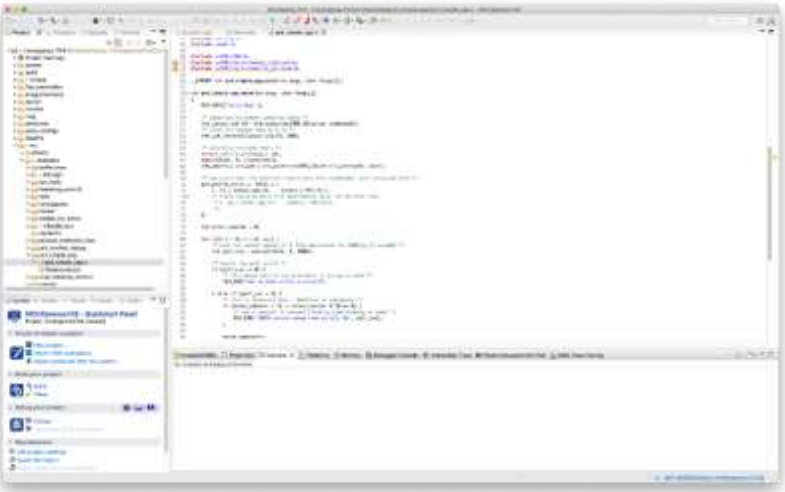
EKF2_MAG_TYPE	Automatic	Type of magnetometer fusion
EKF2_MAG_YAWLIM	0.25 rad/s	Yaw rate threshold used by automatic selection of magnetometer fusion method. This parameter is used when the magnetometer fusion method is set auton
EKF2_MIN_OBS_DT	20 ms	Minimum time of arrival delta between non-IMU observations before data is downsampled. Baro and Magnetometer data will be averaged before downsampli
EKF2_MIN_RNG	0.10 m	Minimum valid range for the range finder
EKF2_MOVE_TEST	1.0	Vehicle movement test threshold
EKF2_NOAID_NOISE	10.0 m	Measurement noise for non-aiding position hold
EKF2_NOAID_TOUT	5000000 uSec	Maximum lapsed time from last fusion of measurements that constrain velocity drift before the EKF will report the horizontal nav solution as invalid
EKF2_OF_DELAY	5.0 ms	Optical flow measurement delay relative to IMU measurements Assumes measurement is timestamped at trailing edge of integration period
EKF2_OF_GATE	3.0 SD	Gate size for optical flow fusion
EKF2_OF_N_MAX	0.50 rad/s	Measurement noise for the optical flow sensor

Software development

PX4 Software Stack

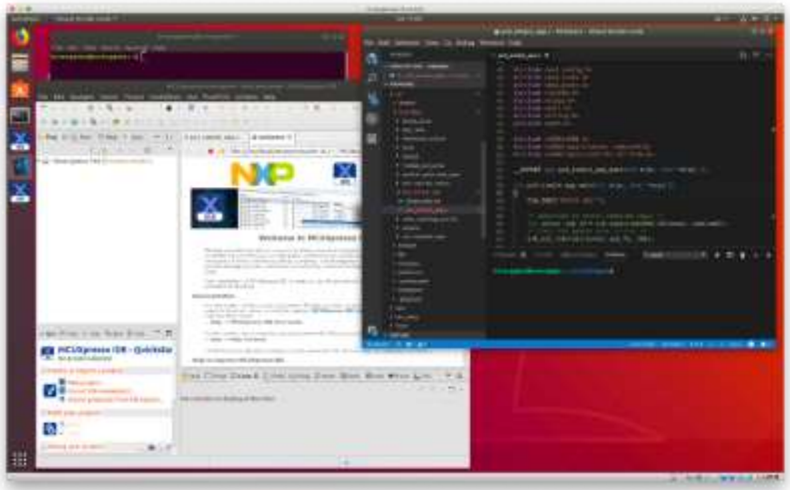


IDE

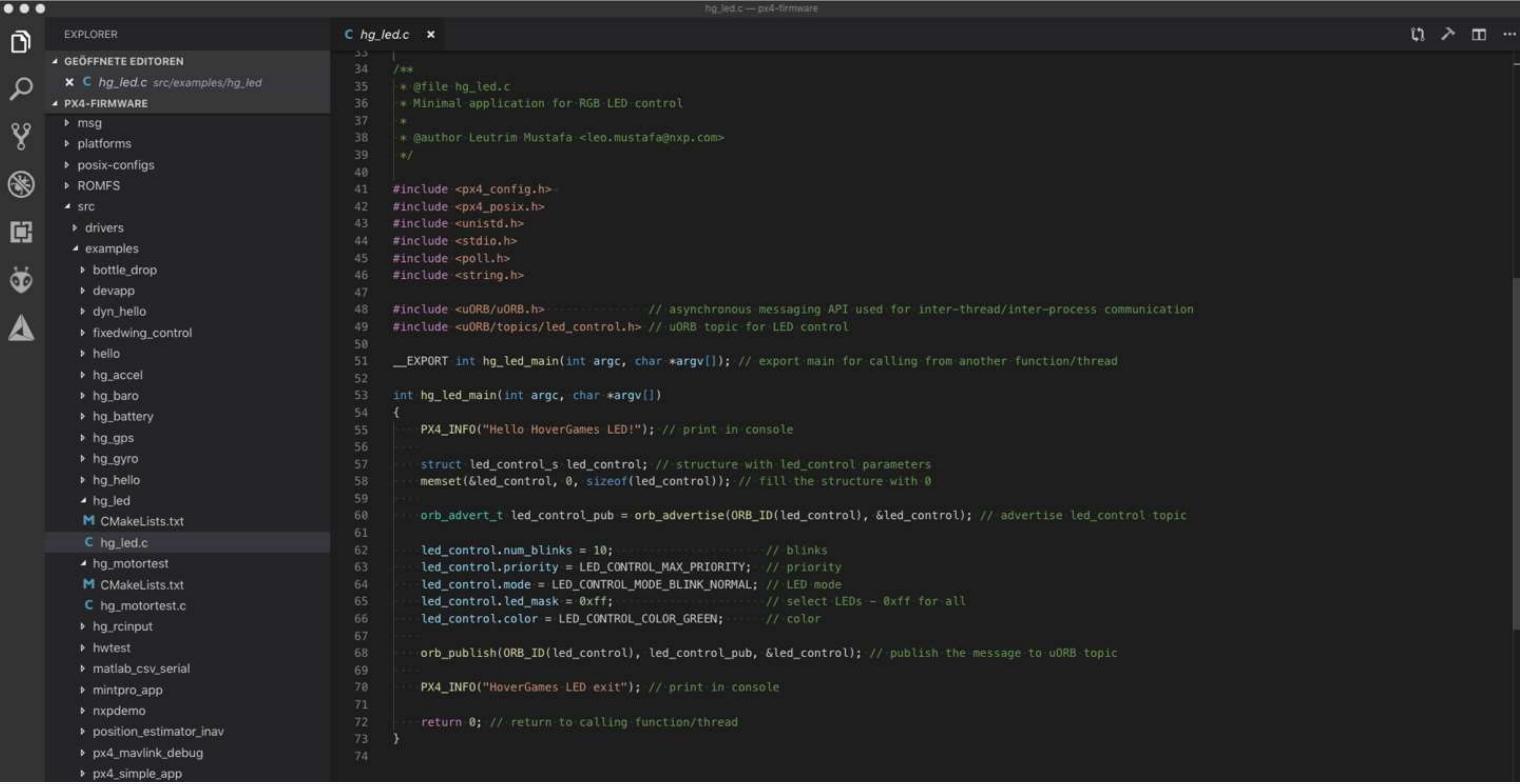


MCUXpresso

Visual Studio Code



VM Image

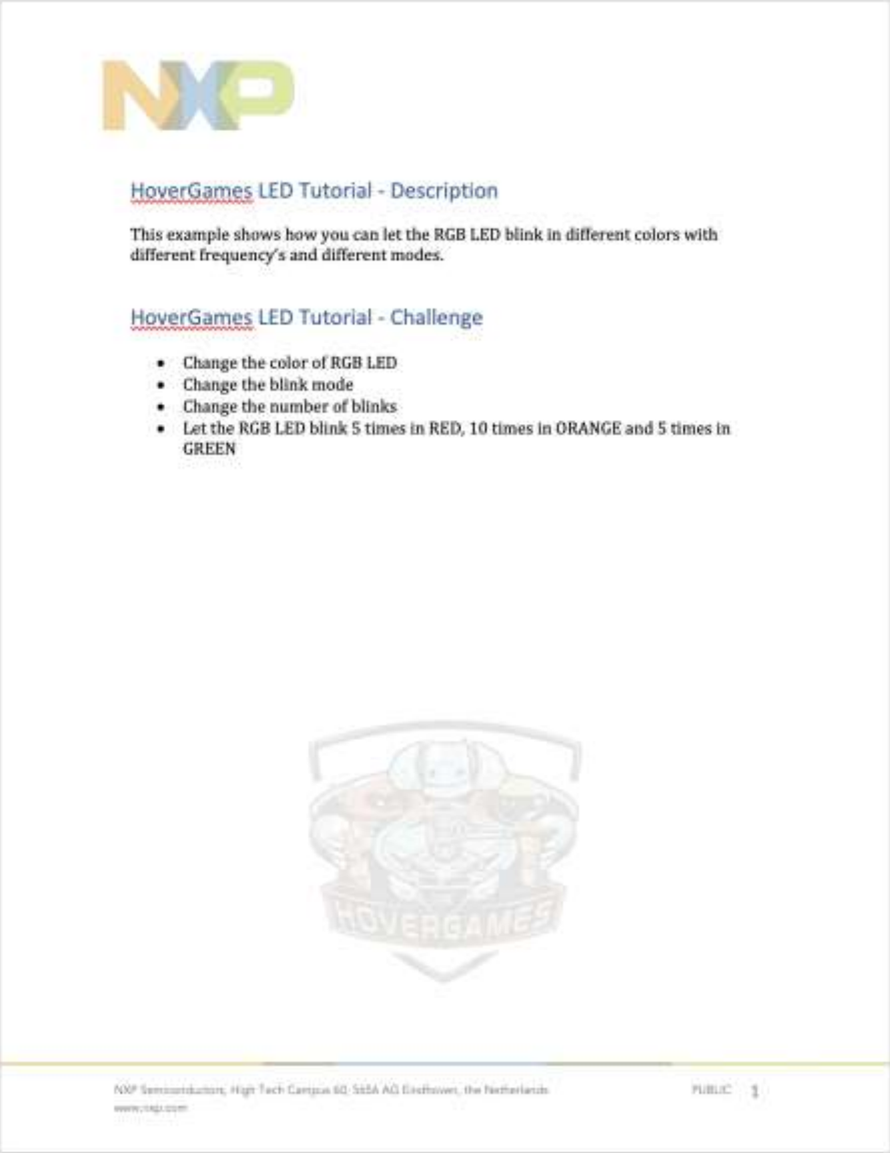


```c
/**
 * @file hg_led.c
 * Minimal application for RGB LED control
 *
 * @author Leutrim Mustafa <leo.mustafa@nxp.com>
 */

#include <px4_config.h>
#include <px4_posix.h>
#include <unistd.h>
#include <stdio.h>
#include <poll.h>
#include <string.h>

#include <uORB/uORB.h>               // asynchronous messaging API used for inter-thread/inter-process communication
#include <uORB/topics/led_control.h> // uORB topic for LED control

__EXPORT int hg_led_main(int argc, char *argv[]); // export main for calling from another function/thread

int hg_led_main(int argc, char *argv[])
{
    PX4_INFO("Hello HoverGames LED!"); // print in console

    struct led_control_s led_control; // structure with led_control parameters
    memset(&led_control, 0, sizeof(led_control)); // fill the structure with 0

    orb_advert_t led_control_pub = orb_advertise(ORB_ID(led_control), &led_control); // advertise led_control topic

    led_control.num_blinks = 10;                      // blinks
    led_control.priority = LED_CONTROL_MAX_PRIORITY;  // priority
    led_control.mode = LED_CONTROL_MODE_BLINK_NORMAL; // LED mode
    led_control.led_mask = 0xff;                      // select LEDs - 0xff for all
    led_control.color = LED_CONTROL_COLOR_GREEN;      // color

    orb_publish(ORB_ID(led_control), led_control_pub, &led_control); // publish the message to uORB topic

    PX4_INFO("HoverGames LED exit"); // print in console

    return 0; // return to calling function/thread
}
```

Overview of tutorials

- hg_accel - Reading accelerometer values from uORB topic
- hg_baro - Reading barometer values from uORB topic
- hg_battery - Reading battery values from uORB topic
- hg_gps - Reading GPS position from uORB topic
- hg_gyro - Reading gyroscope values from uORB topic
- hg_hello - Print hello world on console
- hg_led - RGB LED control
- hg_motortest - Application for motor testing
- hg_rcinput - Reading RC input values from uORB topic

Tutorials



HoverGames LED Tutorial - Description

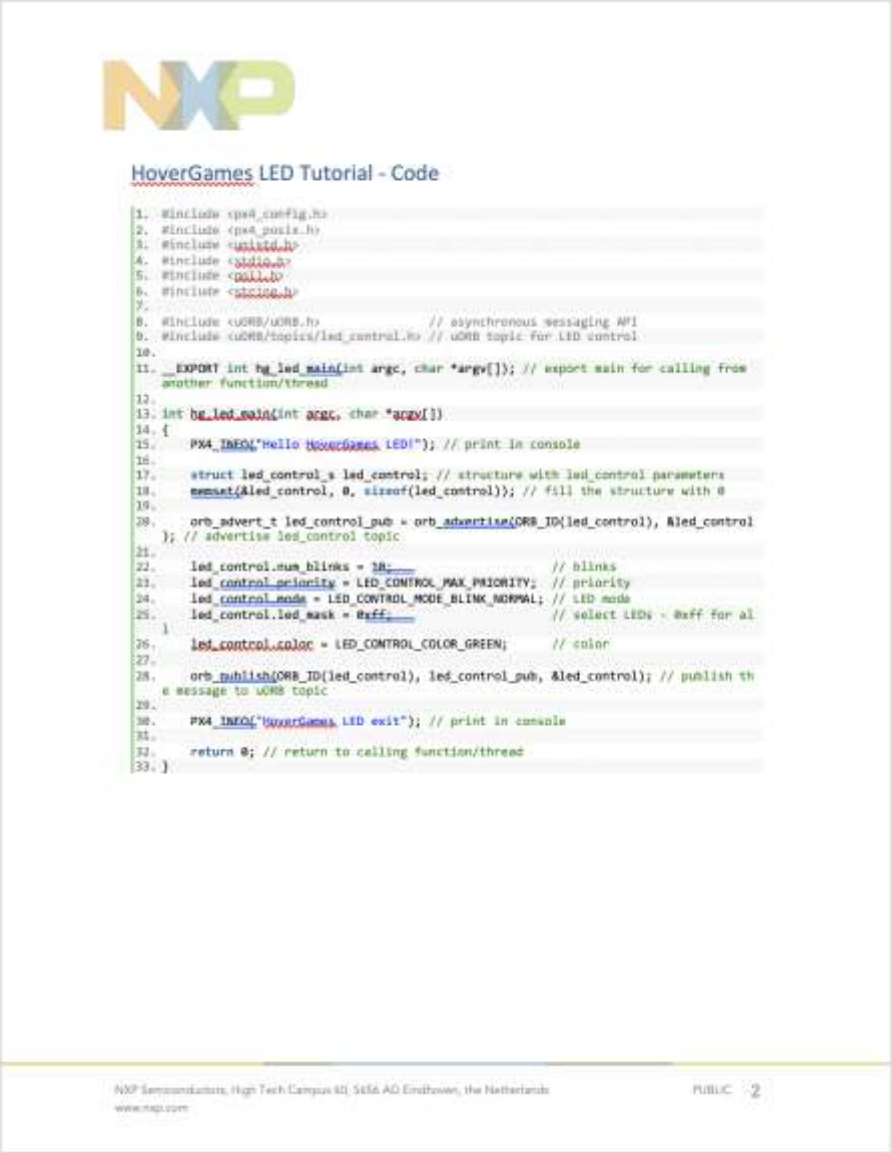
This example shows how you can let the RGB LED blink in different colors with different frequency's and different modes.

HoverGames LED Tutorial - Challenge

- Change the color of RGB LED
- Change the blink mode
- Change the number of blinks
- Let the RGB LED blink 5 times in RED, 10 times in ORANGE and 5 times in GREEN

NXP Semiconductors, High Tech Campus 60, 5656 AG Eindhoven, the Netherlands
www.nxp.com

PUBLIC 1



HoverGames LED Tutorial - Code

```
#include <px4_config.h>
#include <px4_posix.h>
#include <unistd.h>
#include <stdio.h>
#include <poll.h>
#include <string.h>

#include <uORB/uORB.h>                        // asynchronous messaging API
#include <uORB/topics/led_control.h> // uORB topic for LED control

__EXPORT int hg_led_main(int argc, char *argv[]); // export main for calling from
another function/thread

int hg_led_main(int argc, char *argv[])
{
    PX4_INFO("Hello HoverGames LED!"); // print in console

    struct led_control_s led_control; // structure with led_control parameters
    memset(&led_control, 0, sizeof(led_control)); // fill the structure with 0

    orb_advert_t led_control_pub = orb_advertise(ORB_ID(led_control), &led_control
); // advertise led_control topic

    led_control.num_blinks = 10;                      // blinks
    led_control.priority = LED_CONTROL_MAX_PRIORITY;  // priority
    led_control.mode = LED_CONTROL_MODE_BLINK_NORMAL; // LED mode
    led_control.led_mask = 0xff;                      // select LEDs - 0xff for al
l
    led_control.color = LED_CONTROL_COLOR_GREEN;      // color

    orb_publish(ORB_ID(led_control), led_control_pub, &led_control); // publish th
e message to uORB topic

    PX4_INFO("HoverGames LED exit"); // print in console

    return 0; // return to calling function/thread
}
```

NXP Semiconductors, High Tech Campus 60, 5656 AG Eindhoven, the Netherlands
www.nxp.com

PUBLIC 2

SECURE CONNECTIONS
FOR A SMARTER WORLD